Department of Computer Science
Series of Publications A
Report A-2020-1

Methods for Learning Directed and Undirected Graphical Models

Janne Leppä-aho

Doctoral dissertation, to be presented for public examination with the permission of the Faculty of Science of the University of Helsinki, in Auditorium B123, Exactum, on January 24th, 2020 at 12 o'clock noon.

University of Helsinki
Finland

Supervisor
Teemu Roos, University of Helsinki, Finland

Pre-examiners
Joe Suzuki, Osaka University, Japan
Wray Buntine, Monash University, Australia

Opponent
Brandon Malone, NEC Laboratories Europe, Germany

Custos
Teemu Roos, University of Helsinki, Finland

Contact information

Department of Computer Science
P.O. Box 68 (Pietari Kalmin katu 5)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://cs.helsinki.fi/
Telephone: +358 2941 911

ISSN 1238-8645
ISBN 978-951-51-5771-3 (paperback)
ISBN 978-951-51-5772-0 (PDF)
Helsinki 2020
Unigrafia

Methods for Learning Directed and Undirected Graphical Models

Janne Leppä-aho

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
janne.leppa-aho@helsinki.fi

PhD Thesis, Series of Publications A, Report A-2020-1
Helsinki, January 2020, 50+84 pages
ISSN 1238-8645
ISBN 978-951-51-5771-3 (paperback)
ISBN 978-951-51-5772-0 (PDF)

Abstract

Probabilistic graphical models provide a general framework for modeling relationships between multiple random variables. The main tool in this framework is a mathematical object called graph which visualizes the assertions of conditional independence between the variables. This thesis investigates methods for learning these graphs from observational data.

Regarding undirected graphical models, we propose a new scoring criterion for learning a dependence structure of a Gaussian graphical model. The scoring criterion is derived as an approximation to often intractable Bayesian marginal likelihood. We prove that the scoring criterion is consistent and demonstrate its applicability to high-dimensional problems when combined with an efficient search algorithm.

Secondly, we present a non-parametric method for learning undirected graphs from continuous data. The method combines a conditional mutual information estimator with a permutation test in order to perform conditional independence testing without assuming any specific parametric distributions for the involved random variables. Accompanying this test with a constraint-based structure learning algorithm creates a method which performs well in numerical experiments when the data generating mechanisms involve non-linearities.

For directed graphical models, we propose a new scoring criterion for learning

Bayesian network structures from discrete data. The criterion approximates a hard-to-compute quantity called the normalized maximum likelihood. We study the theoretical properties of the score and compare it experimentally to popular alternatives. Experiments show that the proposed criterion provides a robust and safe choice for structure learning and prediction over a wide variety of different settings.

Finally, as an application of directed graphical models, we derive a closed form expression for Bayesian network Fisher kernel. This provides us with a similarity measure over discrete data vectors, capable of taking into account the dependence structure between the components. We illustrate the similarity measured by this kernel with an example where we use it to seek sets of observations that are important and representative of the underlying Bayesian network model.

Computing Reviews (2012) Categories and Subject Descriptors:

Computing methodologies → Machine learning
Mathematics of computing → Probability and statistics
Mathematics of computing → Bayesian networks
Mathematics of computing → Markov networks

General Terms:
machine learning, probabilistic graphical models, model selection

Additional Key Words and Phrases:
Bayesian networks, Markov networks, Bayesian statistics, information theory, pseudo-likelihood, mutual information, normalized maximum likelihood, Fisher kernel

Acknowledgements

I would like to thank my supervisor Professor Teemu Roos for all the guidance and support during my PhD studies. I really appreciate the free and inspiring working environment that has been present throughout my PhD journey.

I am grateful to pre-examiners, Professors Joe Suzuki and Wray Buntine, for taking time to go through this manuscript carefully and providing comments at short notice. I would also like to thank Dr Brandon Malone for agreeing to be the opponent in my defence.

A big thanks goes to all the co-authors of the joint publications included in this thesis. I would especially like to thank Dr Tomi Silander for sharing great ideas and collaboration that resulted in two publications.

I would like to thank all the members, past and present, of the "Information, Complexity and Learning" research group: Jussi Määttä, Yuan Zou, Ville Hyvönen, Elias Jääsaari, Teemu Pitkänen, Jukka Kohonen, Ioanna Bouri, Santeri Räisänen, Sotiris Tasoulis, Yang Zhao and Quan (Eric) Nguyen.

I acknowledge the financial support from Doctoral Programme in Computer Science (DoCS) and Academy of Finland (COIN CoE and project TENSORML).

Finally, my heartfelt thanks go to all my friends and family. My parents, Jaakko and Eija-Liisa, thank you for always being there and providing support and encouragement.

And Aurora, your unconditional love and support has been the biggest thing that kept me going through this journey.

Helsinki, December 2019
Janne Leppä-aho

Contents

Original publications

This thesis is based on the following publications. They are reprinted at the end of the thesis.

I. J. Leppä-aho, J. Pensar, T. Roos, and J. Corander. Learning Gaussian graphical models with fractional marginal pseudo-likelihood. *International Journal of Approximate Reasoning*, 83:21 – 42, 2017.

II. J. Leppä-aho, S. Räisänen, X. Yang, and T. Roos. Learning non-parametric Markov networks with mutual information. In V. Kratochvíl and M. Studený, editors, *Proceedings of the Ninth International Conference on Probabilistic Graphical Models*, volume 72 of *Proceedings of Machine Learning Research*, pages 213–224, Prague, Czech Republic, 11–14 Sep 2018. PMLR.

III. T. Silander, J. Leppä-aho, E. Jääsaari, and T. Roos. Quotient normalized maximum likelihood criterion for learning Bayesian network structures. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 948–957, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

IV. J. Leppä-aho, T. Silander, and T. Roos. Bayesian network Fisher kernel for categorical feature spaces. Accepted for publication in *Behaviormetrika*, 2019.

Author contributions:

Article I: JL formulated and proved the consistency results, performed the experiments and wrote the majority of the paper.

Article II: Based on the idea and preliminary work by SR, XY and TR, JL implemented the method, performed the experiments, and wrote the paper.

Article III: JL proved the consistency and the regularity of the method, wrote the corresponding sections, and took part in analyzing the experimental results.

Article IV: JL proved the invariance theorem, implemented the method, performed the experiments and participated in writing the paper.

None of the articles have been included in any other theses.

Chapter 1

Introduction

> I don't know what that would be — I like graphs. And that doesn't mean I'm going to try to build one. I'm looking for something more like the following graphs.
>
> An excerpt from the output of *GPT-2 language model* built by Adam King when inputted: "This is a nice quote for a computer science PhD thesis:".

This chapter provides a gentle introduction to probabilistic graphical models. Graphical models have been around already for couple of decades and there exists several thorough overviews regarding the subject [25, 29, 63]. We scratch the surface here, going through the general framework briefly and then turn our focus to the main question addressed in this thesis, the problem of learning the structures of graphical models from a set of data. We conclude the chapter by describing the outline for the rest of the chapters and summarize the contributions of the original articles forming this thesis.

1.1 Probabilistic graphical models

Probabilistic graphical models provide a convenient tool for representing and analyzing complex probability distributions over multiple random variables. The graphical models considered in this thesis consist of two main building blocks: 1) a collection of random variables whose joint distribution we are interested in modeling; and 2) a graph, a mathematical object that contains

one node for each random variable and a set of edges between these nodes. The edges in the graph can be undirected or directed, depending whether we are considering Markov networks (equivalently undirected graphs) or Bayesian networks (directed acyclic graphical models). Regardless of which of these two model classes we are using, the graph depicts relationships between the variables by encoding certain assertions of conditional independence that are present in the joint distribution. As such, the graph provides a visual representation of the dependence structure, making some properties of the complex multivariate distribution readily available at a glance, and what is more, the graph can be often used to decompose the distribution into smaller components that admit easier interpretation and are more tractable to handle computationally.

By providing a very general framework for representing multivariate distributions, graphical models are applied widely in the fields of machine learning and statistics. For instance, the structures of many commonly encountered models in these areas are easily described using a directed graphical model. Some classical examples include [4]: mixture models, (probabilistic) principal component analysis, factor analysis, naive Bayes, and also, to name a more recent example, variational auto-encoders [24].

When treating graphical models, the discussion separates intrinsically in three distinct topics [25]: *representation*, *inference*, and *learning*. We started our discussion by noting how graphical models are a tool for representing complex probability distributions. Then, having specified the distribution with the help of a graph, we might want to use our model to answer probabilistic queries regarding some of the variables. This is the part where the inference comes into play and often the graph structure is in the key role by determining how efficiently we can run our inference procedures. However, the part we are dedicating most of the focus in this thesis is the learning which tackles the problem of coming up with the graphical model in the first place.

1.2 Learning graphical models

In the learning part, we are given a set of observed data on some variables and our goal is to learn the graph that specifies the dependence structure among variables. This part is called *structure learning*. Additionally, we might also be interested in learning the parameters for this structure which might involve a separate step or happen at the same time with the structure learning, depending on the used method.

The motivation for structure learning might be knowledge discovery: by learning the graph, we gain insight on the variables in our domain as we know how they depend on each other. For instance, directed graphical models have been used to model potential relationships in protein signaling networks [47] and brain-region connectivity using fMRI data [19]. The graph learning might also be motivated as an initial step for the subsequent prediction or inference tasks. For instance, in a classification task, we are interested in predicting the value for a class variable given the values for predictor variables. Examples of graphical models that are learned with the classification in mind include: naive Bayes, tree augmented naive Bayes [13] and limited dependence Bayesian network [48] classifiers.

In general, the learning of graphical models is a challenging problem. For the undirected graphs, the number of possible graphs increases exponentially with respect to the number of variables. In case of directed acyclic graphs, there exists even more possible graph structures for a given set of variables.

The approaches for learning graphical models are commonly divided in two categories. The *score-based* methods cast the learning as a model selection problem. In this approach, each graph implies a model over our data that we can score using some criterion. The learning reduces thus to finding the highest scoring structure.

The *constraint-based* methods approach the problem from a different angle. They utilize the fact that the edges in the graphical model imply a set of conditional independence statements over the variables. Based on the observed data, we can test whether a particular independence statements are likely to hold with the help of the well-known machinery from statistical literature called *hypothesis testing*.

1.3 Outline of the thesis

This thesis studies the learning of graphical models in different scenarios. Without yet going much into the details, we will be tackling the following kinds of questions:

- Assume that we have a large number of variables and possibly a small amount of observations. Assume further that we model the data with a multivariate normal distribution. How can we learn an undirected graph describing the dependence structure of variables?

- What about if we do not want to make restrictive assumption of multivariate normality? Here, we focus on situations where the number of variables is relatively small but the variables might depend on each

other in a complex, non-linear way. How do we learn an undirected graph in this situation?

- Assume now that the variables are categorical and we want to learn a directed acyclic graph. Adopting a score-based approach, there are plenty of scoring functions to choose from, each with its own virtues and vices. Can we come up with a new one that would avoid at least some of the shortcomings of the previous approaches?

In addition to problems related to learning, we also consider one application where we make use of the learned Bayesian network. We start with a completely specified Bayesian network model over categorical variables. Using this network we address the following problem:

- How can we measure similarity among data vectors with the aid of our Bayesian network? In general, being able to compute similarity for categorical data would give rise to myriad of applications but what would be the thing that this model induced similarity is the most suitable for?

The rest of the thesis is organized as follows. After providing the summaries of the original articles in this thesis, we move on to the Chapter 2 where we dive into the world of graphical models in an increased level of formality, providing notations and concepts needed to grasp the content of the remaining chapters. Then the two subsequent chapters treat the content of the original articles: Chapter 3 is based on original articles I and II with an underlying theme of learning undirected graphical models, and articles III and IV are discussed in Chapter 4, where the main topic is directed acyclic graphs. Chapter 4 is not purely focused on learning as we also present an application on how to make use of the learned networks. Finally, conclusions are presented in Chapter 5.

1.4 Main contributions

In this section, we briefly summarize the main contributions of the original articles.

Article I: Learning Gaussian graphical models with fractional marginal pseudo-likelihood. We propose a new scoring criterion for learning the dependence structure of the Gaussian graphical model. The criterion makes use of pseudo-likelihood in order to express the approximative marginal likelihood for any undirected graph in closed form. It is

applicable in high dimensional settings and also shown to be consistent. In the experiments, we pair the criterion with an efficient greedy algorithm and evaluate the performance of the method against the leading methods for learning Gaussian graphical models.

Article II: Learning non-parametric Markov networks with mutual information. We propose a method for learning Markov network structures without restricting the learned network to belong to any specific family of parametric distributions. The method makes use of a non-parametric estimator of conditional mutual information to test independence assertions. The resulting independence test is combined with a constraint-based learning algorithm, and shown to work well in the settings where the relationships between variables involve non-linearities.

Article III: Quotient normalized maximum likelihood criterion for learning Bayesian network structures. We present a new scoring criterion for learning Bayesian network structures in the discrete setting. The criterion is motivated as an approximation to information theoretic quantity called Normalized Maximum Likelihood. We discuss the theoretical properties of the score and compare it empirically to other commonly used criteria.

Article IV: Bayesian network Fisher kernel for categorical feature spaces. In this article, we derive a closed form expression for a Fisher kernel derived from a general Bayesian network model over categorical variables. We show that the resulting kernel is invariant for Bayesian networks expressing the same assertions of conditional independence. The Bayesian network Fisher kernel is studied empirically in experiments where the aim is to use the kernel to gain insight into the underlying Bayesian network.

Chapter 2

Preliminaries: graphical models

This chapter introduces the notation and basic concepts related to graphical models. After the overview, we start with the Bayesian networks and then move on to the undirected graphical models. With the general notation fixed, we then discuss the structure learning of graphical models more in detail.

2.1 General notation

Let $X = (X_1, \ldots, X_d)^T$ denote a d-dimensional random vector. Let $G = (V, E)$ denote a graph, where $V = \{1, \ldots, d\}$ is the set of nodes and $E \subset V \times V$ denotes the set of edges. We associate each random variable X_i with the node $i \in V$, $i = 1, \ldots, d$. The terms node and variable are used interchangeably. We use X_A, $A \subset \{1, \ldots, d\}$ to refer to a subvector of X restricted to variables in the set A.

The edges in the graph can be directed or undirected. In this thesis, we treat only graphs that include one of the aforementioned type at a given time. We say that there exists a directed edge from X_i to X_j, denoted also as $X_i \to X_j$, if and only if $(i, j) \in E$. If $(i, j) \in E$, variable X_i is said to be a parent of X_j. In case of an undirected edge between X_i and X_j, $X_i - X_j$, the edge set E contains tuples (i, j) and (j, i).

2.2 Directed graphical models

This section considers Bayesian networks, or equivalently, directed acyclic graph (DAG) models. Assume that G is a directed acyclic graph, implying that all the edges are directed and there does not exist any directed cycle in G. Let $p(X)$ denote the joint distribution of X. The graph G encodes a set

of conditional independence assertions between the components of X that can be characterized with *Markov properties* [29]. The *local directed Markov property* states that each variable X_i is independent of its *non-descendants* given its parents $X_{\pi(i)}$. A variable X_i is non-descendant of X_j if there does not exist a directed path from X_j to X_i. We use $\pi(i) = \{j \mid (j,i) \in E\}$ to denote the parent set of variable X_i. The local Markov property is equivalent to the following factorization of $p(X)$ according to G:

$$p(X) = \prod_{i=1}^{d} p(X_i \mid X_{\pi(i)}). \tag{2.1}$$

This decomposition (2.1) to a product of conditional distributions is sometimes referred as the chain rule for Bayesian networks. It allows us to parametrize the joint distribution $p(X \mid \theta)$ conveniently through defining local parameters θ_i for each conditional distribution $p(X_i \mid X_{\pi(i)}, \theta_i)$.

For instance, assuming our variables are continuous, we can take conditional distributions to be linear Gaussian

$$X_i \mid X_{\pi(i)} = x_{\pi(i)} \sim \mathcal{N}(\boldsymbol{w}_i^T x_{\pi(i)}, \sigma_i^2), \tag{2.2}$$

with the local parameters θ_i being now the vector of edge strengths $\boldsymbol{w}_i$ which defines the mean of the distribution and the conditional variance σ_i^2. If we define the conditional distributions using (2.2), the joint over X will be a multivariate normal $\mathcal{N}_d(\mathbf{0}, \Sigma)$, where Σ can be determined from the collection of the local parameter sets [38].

Assume next that the variables are discrete, meaning that each X_i can take values from 1 to $r_i \in \mathbb{N} \setminus \{0, 1\}$ (variables are treated categorical even though encoded using integers). In this case, it is common to parametrize the DAG structure using conditional probability tables by enumerating all the possible combinations of values for parents $X_{\pi(i)}$ of X_i, and defining

$$\theta_{ijk} = P(X_i = k \mid X_{\pi(i)} = j), \tag{2.3}$$

where $i \in \{1, \ldots, d\}$, $k \in \{1, \ldots, r_i\}$ and $j \in \{1, \ldots, q_i\}$ with q_i denoting the number of possible parent combinations and $X_{\pi(i)} = j$ meaning that the parent variables take values according to the j^{th} configuration. For each i, the local distribution contains $q_i \cdot (r_i - 1)$ free parameters, as the probabilities need to sum to one for any given parent combination. In other words, we model each variable with one categorical/multinomial distribution for each possible assignment of its parent variables. The parameter vectors related to these conditional distributions are denoted by $\theta_{ij} = \{\theta_{ijk} \mid k = 1, \ldots, r_i\}$.

This allows us to express the likelihood of θ for a single data point $X = x$ as

$$p(x \mid \theta, G) = \prod_{i=1}^{d} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{\mathbb{I}(x_i=k,\ x_{\pi(i)}=j)}, \tag{2.4}$$

where the indicator function $\mathbb{I}(x_i = k,\ x_{\pi(i)} = j) = 1$ if $x_i = k$ and $x_{\pi(i)} = j$; and $\mathbb{I}(x_i = k,\ x_{\pi(i)} = j) = 0$, otherwise.

With directed graphical models it is possible that two different DAG structures G_1 and G_2 represent exactly the same assertions of conditional independence. The graphs G_1 and G_2 are then said to independence equivalent. In case of the above two examples where the conditional distributions are linear Gaussian or multinomial, independence equivalence implies also that the two DAGs G_1 and G_2 represent the same set of joint distributions over X [14].

2.3 Undirected graphical models

The undirected graphical models (Markov networks), as the name implies, represent the independence structure for a set of random variables with the help of an undirected graph G. The possible sets of conditional independence assertions that can be represented with undirected graphs are generally different from those we can represent using DAGs. However, there exists a class of graphs called *chordal* or *decomposable* graphs consisting of independence structures that are equally well representable using DAGs or undirected graphs.

Also in the undirected case, we can characterize the independence assumptions using similar Markov properties as those mentioned in the directed case. Since the edges do not have directions, we do not have concepts like "parent" or "child" and the properties admit maybe a bit simpler form.

Let $mb(i)$ denote the Markov blanket of variable X_i. The set $mb(i)$ includes all nodes that are connected to i by an edge in G. Now, the *pairwise, local*, and *global Markov properties* can be stated as follows:

1. If there is no edge between X_i and X_j, the variable X_i is independent of X_j given the remaining variables.

2. Given its Markov blanket, each variable X_i is conditionally independent of all the remaining variables.

3. For disjoint subsets of nodes, $A, B, C \subset V$, it holds that X_A is conditionally independent of X_B given X_C, if C separates A and B in the graph G.

These three properties are equivalent assuming the positivity of the distribution $p(X)$ [29].

Even though the independence assertions are somewhat easier to look up from an undirected graph, the distribution does not in general factorize into as intuitive components as the conditional probability distributions in case of DAGs. With Markov networks, the distribution factorizes over the *maximal cliques* of the graph. A clique is a set of nodes in a graph such that each node is connected by an edge to all the remaining nodes in the clique. Moreover, a clique is maximal if there does not exist a node in the graph which could be added to it while still satisfying the clique definition. Let $\mathcal{C}$ denote the set of maximal cliques in G. Now,

$$p(X) = \frac{1}{Z} \prod_{C \in C} \phi(X_C), \tag{2.5}$$

where $\phi(X_C)$ are called clique potentials which map the values of random variables to (usually) strictly positive values. The term $Z^{-1} = \sum_x \prod_{C \in C} \phi(X_C = x_C)$ is the normalization constant which guarantees that the product defines a valid distribution. In case of continuous variables the sum would be replaced by an integral. As the requirement for a mapping to be a potential function is rather loose, we generally lose the ability to interpret these as probability distributions.

One specific class of undirected graphical models encountered also later in this thesis is Gaussian graphical models. In Gaussian graphical models, we have an undirected graph G and a random vector X following a multivariate normal distribution $\mathcal{N}_d(\mathbf{0}, \Omega^{-1})$, where the matrix Ω is the inverse of covariance, aka the precision matrix. Due to the properties of multivariate normal distribution, the graph structure is easily read from the precision matrix. To be more precise, X_i is conditionally independent of X_j given the rest of variables if, and only if $\Omega_{ij} = 0$. This means that there is no edge between these variables in the graph G. In other words, in undirected Gaussian graphical models, the graph structure is visible in the zero pattern of Ω but otherwise the elements are unrestricted as long as the resulting matrix is positive definite.

2.4 Graphical model structure learning

In this section, we define the problem of learning the structure of graphical model a bit more carefully, review the outlines of the common strategies when tackling this problem and mention some related concepts we will encounter later in the thesis.

2.4.1 Defining the problem

In every article included in this thesis, we are either proposing a method for solving, or just encountering the following problem: we are given a data matrix $\boldsymbol{X} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n)$ consisting of d-dimensional observations $\boldsymbol{x}^j$. Depending on the problem, components x_i^j might be real numbers or integers representing the values of continuous or categorical variables, respectively. We have the underlying assumptions that $\boldsymbol{x}^j$ are realizations of independent and identically distributed (i.i.d.) random variables following a distribution whose dependence structure obeys the Markov properties implied by graph G. Based on the observational data $\boldsymbol{X}$, our aim is to recover G.

Next, we will discuss the score-based methods and how they approach this problem.

2.4.2 Score-based learning

The score-based approach treats the problem of learning a graph structure as a model selection problem. With d variables, we have a finite amount of possible graphs G to choose from, and each graph is associated with a model $\mathcal{M}_G$ for our data. Model

$$\mathcal{M}_G = \{\ p(\ \cdot \mid G, \theta) \mid \theta \in \Theta_G \subseteq \mathbb{R}^k\ \}$$

means here a set of probability distributions for data $\boldsymbol{X}$ that have a common functional form depending on G, and that are indexed by some k-dimensional parameter vector θ.

A scoring function is mapping which returns a scalar value for every admissible input of $\boldsymbol{X}$ and G. We can interpret this scalar value to describe how well our model fits the observed data. Making our models more complex, we can naturally fit the data better, thus scoring functions usually include a term that takes into account the complexity of the model, making the total score a trade-off between the goodness-of-fit and model complexity penalty.

There exists a wide variety of possible scoring functions devised under different assumptions. In this thesis, we will discuss scoring functions that draw their motivation from Bayesian statistics [2, 15] and Minimum Description Length (MDL) principle [17, 44].

With the scoring function given, the learning problem becomes an optimization problem over the possible graph structures. When learning DAGs, most of the scoring functions are *decomposable*, meaning that the score for the whole network can be expressed as a variable-wise product (or a sum) where the i^{th} local term depends only on the data on X_i and its parents $X_{\pi(i)}$. This allows the score to be optimized by traversing the

space of graphs with the help of local updates that affect only a couple of terms in the decomposition. Decomposability is not that often encountered when learning undirected graphs, although we will see a counter-example in Chapter 3.

Another theoretical property that we will discuss later in the thesis when proposing new scoring functions is *consistency*. Consistency roughly guarantees that our scoring function will eventually give the highest score to the true generating graph as we let our data size n tend to infinity.

2.4.3 Constraint-based learning

In the constraint-based approach, we make use of the fact that the graph defines a certain set of independence assumptions over the variables in our domain. These are the Markov properties we reviewed earlier. Based on the observed data, we can then perform a series of queries to try to find Markov properties that hold and parse our graph together from these results. In practice, a single query is a conditional independence test which can be formulated as a hypothesis test. For an overview on hypothesis testing, see Casella and Berger [6].

The result of performing a hypothesis test is a yes/no answer, which usually indicates a presence or absence of an edge in the graphical model. Various algorithms have been proposed in order to be able to learn the network structures efficiently without resorting to testing every possible assertion of conditional independence.

For instance, in Chapter 3, we will use an algorithm that makes use of the local Markov property when learning an undirected graph. The idea is to find the Markov blanket for each variable, eq. the smallest set of variables that renders the variable conditionally independent of the remaining ones.

Chapter 3

Learning undirected graphical models

In this chapter, we discuss two methods for learning undirected graphical models with continuous variables that differ greatly in modeling assumptions they are making about the underlying data generating distribution. In the first approach, we model our data with a multivariate normal distribution, and using an approximation to the likelihood called *pseudo-likelihood*, we derive a computationally attractive scoring function, titled as fractional marginal pseudo-likelihood (FMPL), that allows us to learn graphs with very large number of variables. In the second part, we devise a test for conditional independence that does not require us to assume any particular parametric form of the distribution for the variables involved. This allows us to identify complex interactions among variables more accurately but the resulting method is also computationally more involved.

3.1 Learning high dimensional Gaussian graphical models

We assume that our data $\boldsymbol{X} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n)$ are i.i.d samples from a multivariate normal distribution $\mathcal{N}_d(\mathbf{0}, \Omega^{-1})$, where the structure of the (positive definite) precision matrix Ω is determined by some undirected graph G^*, which describes the dependency structure of the generating distribution. In other words, we are modeling our data using the Gaussian graphical model (GGM), mentioned in Chapter 2.

3.1.1 Bayesian learning of GGMs

We adopt a score-based approach to learning and make use of the Bayesian framework in deriving our scoring function. To briefly summarize, the Bayesian approach requires us to first construct a joint probability model for all the relevant quantities (observed and unobserved) in our problem domain, which involves expressing our beliefs on unobserved quantities by assigning prior distributions over them. Next task is to compute the posterior distribution for quantities under interest, and then finally use this posterior as the basis of decision making. In the context of score-based learning, Bayesian approach boils down to finding the graph with the highest posterior probability $p(G \mid \boldsymbol{X})$. The posterior for G can be written as follows:

$$p(G \mid \boldsymbol{X}) \propto p(\boldsymbol{X}, G) = p(\boldsymbol{X} \mid G) \cdot p(G), \tag{3.1}$$

where $p(G)$ is the prior probability for G and $p(\boldsymbol{X} \mid G)$ denotes the marginal likelihood. In the first proportionality, we omitted the term $p(\boldsymbol{X})$ as this is constant for every G, and can be ignored when we are only interested in finding the G with the maximum posterior probability. Marginal likelihood is the only data dependent term in Eq. (3.1). It is defined as follows:

$$p(\boldsymbol{X} \mid G) = \int_{\theta \in \Theta_G} p(\boldsymbol{X} \mid \theta, G)\ p(\theta \mid G)\ d\theta, \tag{3.2}$$

where Θ_G denotes the set of possible parameter values under G. Assuming uniform prior over graphs, the structure learning problem reduces to finding the graph with the highest marginal likelihood. However, even with our Gaussian assumption, there are several problems when trying to evaluate the marginal likelihood integral under a general graph structure G:

1. We need to be able to specify the parameter prior $p(\theta \mid G)$ for any given network structure. Eliciting the prior distributions subjectively might quickly become a daunting task as the number of variables increases.

2. Marginal likelihood is easily[1] evaluated in closed form only if the underlying graph is chordal [10].

3. Methods based on numerical approximations of marginal likelihood tend to get computationally very demanding as the number of variables grows. For instance, Wang and Li [62] report that approximating the

[1]Recent work [60] shows that there exists, in principle, a way to evaluate marginal likelihood in closed form for a general graph structure.

marginal likelihood of a 100 node graph using Monte Carlo integration would take approximately two days[2].

However, evaluating the marginal likelihood for a Gaussian *directed* graphical model is easier. In addition, there exists work on the objective comparison of Gaussian DAGs [8] which helps us to deal with the difficulty of eliciting the prior distributions. We will next review these results briefly and then describe how we can use pseudo-likelihood to connect the framework developed for the Gaussian DAGs to our problem of learning the undirected graph structures.

3.1.2 Objective comparison of Gaussian DAGs

Consonni and La Rocca [8] describe a framework for computing marginal likelihoods objectively for any Gaussian DAG structures. Objectivity is attained using an uninformative, usually also an improper, prior over the model parameters. Their methodology is based on a more general framework introduced by Geiger and Heckerman [14] which requires specifying only a single prior for the parameters of the complete DAG model (the precision matrix Ω for a model implying no assertions of conditional independence). If certain regularity assumptions are satisfied, this allows one to obtain the marginal likelihood for any Gaussian DAG D using the formula

$$p(\boldsymbol{X} \mid D) = \prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}, D_c) = \prod_{i=1}^{d} \frac{p(\boldsymbol{X}_{fa(i)} \mid D_c)}{p(\boldsymbol{X}_{\pi(i)} \mid D_c)}, \tag{3.3}$$

where D_c refers to a complete DAG model, for which we have specified the prior distribution, $fa(i)$ is the shorthand notation for $\pi(i) \cup \{i\}$, and the terms appearing on the right-hand side of (3.3) are marginal likelihoods corresponding to data on subvectors of X under D_c. For the parameters of the full DAG model, Ω, Consonni and La Rocca use a default, uninformative prior of the form

$$p(\theta \mid D_c) = p(\Omega) \propto |\Omega|^{(\alpha-d-1)/2},$$

where $|\cdot|$ denotes the matrix determinant and α is a free parameter which we will take to be $\alpha = d - 1$, yielding $p(\Omega) \propto |\Omega|^{-1}$. In order to cope with the possible difficulties arising from the use of an improper prior, Consonni and La Rocca apply fractional Bayes factors [40].

In the fractional Bayes framework, we use a fraction $0 < b < 1$ of the likelihood, $p(\boldsymbol{X} \mid \theta)^b$, to update the improper prior $p(\theta)$ to a proper posterior,

[2]Using a quad-CPU 3.33GHz desktop computer.

called the fractional prior. This fractional prior is then paired with the $1-b$ fraction of the likelihood when computing the marginal likelihood. In our setting, with $\alpha = d-1$, we can take $b = 1/n$, which results the fractional prior over Ω being a proper, data dependent Wishart distribution. This then allows one to express the terms in Eq. (3.3) in closed form.

3.1.3 FMPL score

Next we will make use of pseudo-likelihood [3] in order to connect the marginal likelihood results of the directed case to the undirected one. Pseudo-likelihood replaces the true likelihood function with an approximation that is computationally more tractable. Using the chain rule, we can always write

$$p(\boldsymbol{X} \mid \theta, G) = \prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_1, \dots \boldsymbol{X}_{i-1}, \theta, G). \tag{3.4}$$

The main trick in pseudo-likelihood is to add more conditioning variables in each term of Eq. (3.4). Denoting $X_1, \dots X_{i-1}, X_{i+1}, \dots, X_d = X_{-i}$, we get the pseudo-likelihood as

$$\prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_1, \dots \boldsymbol{X}_{i-1}, \theta, G) \approx \prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_{-i}, \theta, G),$$

and using the Markov properties implied by G, this simplifies further to

$$\prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_{mb(i)}, \theta).$$

Now replacing the likelihood in (3.2) with the above pseudo-likelihood, and assuming that the marginal likelihood integral factorizes over parameter sets θ_i related to conditional distributions $p(X_i \mid X_{mb(i)})$, we obtain *marginal pseudo-likelihood* as

$$\hat{p}(\boldsymbol{X} \mid G) \equiv \prod_{i=1}^{d} \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{mb(i)}) = \prod_{i=1}^{d} \int_{\theta_i} p(\boldsymbol{X}_i \mid \boldsymbol{X}_{mb(i)}, \theta_i)\ p(\theta_i)\ d\theta_i. \tag{3.5}$$

Marginal pseudo-likelihood was originally introduced in the context of discrete undirected graphical models by Pensar et al. [42]. We refer to terms $\hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{mb(i)})$ as the local marginal pseudo-likelihoods.

Marginal pseudo-likelihood bears a resemblance to the marginal likelihood under a DAG model as seen by comparing Eq. (3.5) to Eq. (3.3). Thus, by using the available closed form formula for (3.3), replacing $pa(i) \to mb(i)$

and re-defining $fa(i) = mb(i) \cup \{i\}$, we obtain fractional marginal pseudo-likelihood as

$$\hat{p}(\boldsymbol{X} \mid G) = \prod_{i=1}^{d} \pi^{-\frac{(n-1)}{2}} \frac{\Gamma\left(\frac{n+p_i}{2}\right)}{\Gamma\left(\frac{p_i+1}{2}\right)} n^{-\frac{2p_i+1}{2}} \left(\frac{|\boldsymbol{S}_{fa(i)}|}{|\boldsymbol{S}_{mb(i)}|}\right)^{-\frac{n-1}{2}}, \tag{3.6}$$

where p_i is the size of the set $mb(i)$, $\boldsymbol{S} = \boldsymbol{X}^T\boldsymbol{X}$ is the unscaled covariance matrix (assuming $\boldsymbol{X}$ has observations on rows), and the notation $\boldsymbol{S}_A$ refers to the submatrix of $\boldsymbol{S}$ restricted to variables in set A. The above score is well-defined if matrices $\boldsymbol{S}_{fa(i)}$ and $\boldsymbol{S}_{mb(i)}$ are positive definite.

3.1.4 Properties of FMPL

The FMPL score defined in the last section is completely free of any tunable hyper-parameters, which is naturally an attractive property. However, the derivation presented in the last section might seem a bit heuristic. To put the score on a firmer ground, we formulate and prove the following theorem in Article I which verifies that FMPL is consistent estimator for the undirected graph structure:

Theorem 3.1 (Theorem 2 in Article I)**.** *Let $X \sim N_d(\mathbf{0}, (\boldsymbol{\Omega}^*)^{-1})$ and $G^* = (V, E^*)$ denote the the undirected graph that completely determines the conditional independence statements between the components of X. Let $\{mb^*(1), \ldots, mb^*(d)\}$ denote the set of Markov blankets, which uniquely define G^*.*

Suppose we have a complete random sample $\boldsymbol{X}$ of size n obtained from $N_d(\mathbf{0}, (\boldsymbol{\Omega}^*)^{-1})$. Then for every $i \in V$, the local fractional marginal pseudo-likelihood estimator

$$\widehat{mb}(i) = \arg\max_{mb(i) \subset V \setminus \{i\}} \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{mb(i)})$$

is consistent, that is, $\widehat{mb}(i) = mb^*(i)$ with probability tending to 1, as $n \to \infty$.

As the Markov blankets define the graph uniquely, Theorem 3.1 guarantees that the true graph will eventually receive the highest score.

Another remarkable property of the FMPL scoring function is that it is decomposable, a property not so often encountered in the context of undirected graphs, and we can optimize it independently for each variable while still guaranteeing the consistency in the limit of infinite data. We will make use of this property in the next section, where we will review a greedy algorithm for optimizing the FMPL score. In Article I, we also provide

theoretical results showing that this greedy algorithm equipped with FMPL score will eventually identify the true Markov blankets when given enough data.

3.1.5 Optimizing the FMPL score

Our approach to optimizing the FMPL score is divided in two steps:

1. We start by finding the Markov blanket for each node independently. This is done by using a greedy algorithm that is similar in spirit to a constraint-based algorithm called interIAMB [58]. The found Markov blankets are combined to two undirected graphs: G_{OR} and G_{AND}.

2. Making further use of the decomposability of the score, we run greedy hill-climbing based on local changes starting from an empty graph. The allowed operations are adding or deleting an edge from the graph. As a further restriction, only edges present in G_{OR} are considered when edges are added. The algorithm terminates after no local change provides an increase in FMPL score. The output graph is called G_{HC}.

To describe the first step more carefully, the algorithm starts from an empty blanket and then adds a node there that results in the highest increase in the score. Successful addition steps are followed by deletion steps, where nodes are removed from the blanket if that increases the score. The algorithm terminates after one unsuccessful addition step.

Even though the FMPL score is consistent, this result applies only asymptotically, and the Markov blankets found in the first step with any finite sample sizes might not be coherent. By coherent, we mean that if we found i to belong to the Markov blanket of j then j should also be in the blanket of i as per definition of an undirected graph. To enforce this property with finite sample sizes, we output two graphs mentioned already above: G_{AND} includes only the edges that were found in both directions during independent Markov blanket searches, and for denser G_{OR} it is enough that edge was found in one direction.

This procedure is suitable for high-dimensional settings as the Markov blanket searches in the first step can be computed completely in parallel. Also, the edge addition and deletion operations involved in the second step are efficient to evaluate due to the decomposability of the score: the score needs to be recomputed only for the two nodes involved in the local change.

A similar two step strategy was used in the context of learning discrete undirected graphs by Pensar et al. [42]. In general, the procedure resembles the two step algorithm for learning DAG structures called Max-Min Hill-Climbing [59]. The main difference is that Max-Min Hill-Climbing uses a

constraint-based algorithm in the first step when learning the undirected skeleton of the network.

3.1.6 On the empirical performance

To evaluate the FMPL method in practice, we compared it to three commonly used methods for learning Gaussian graphical models: graphical lasso (glasso) [12, 64], neighbourhood selection (NBS) [36], and Sparse Partial Correlation Estimation (SPACE) [41]. The common denominator in all the aforementioned methods is that they use ℓ_1-penalty in their objective functions in order to promote sparsity in the solutions. In the experiments, we also included an additional sparsity promoting prior on the sizes of Markov blankets to FMPL score. A more detailed description of the prior is found in Article I.

The different methods were evaluated in two tasks: structure learning and prediction. In the structure learning experiments, the graphs found by the considered methods were compared to the ground truth graph using Hamming distance (the number of edges to be added and deleted in order to obtain the true graph). In the prediction experiments, the task was to predict a value for a variable given the values of all the other variables based on a model learned from training data.

To briefly summarize the conclusions from the experiments: in terms of structure learning, the AND and HC graphs outputted by FMPL method were generally closer to the true generating graph than the ones produced by the competitors. However, in terms of prediction, the compared methods performed quite similarly and one clear winner for all the settings was hard to pick. In the prediction experiments, OR graph seemed to outperform the other graphs outputted by FMPL.

To highlight some of the numerical results, we show in Figures 3.1 and 3.2 the results for the different methods in structure learning experiments with synthetic data. The ground truth graphs had the number of nodes ranging from 64 from 1024 with the corresponding number of edges ranging from 78 to 1248. The inverse of covariance was randomly created with the zero pattern implied by the graph, and the data was sampled from the multivariate normal distribution determined by the inverse of covariance. Figure 3.1 shows the comparison of AND graph of FMPL and the ℓ_1-based methods. We can see that generally, the AND method performs the best with NBS on tail. Figure 3.2 shows the comparison between the different graphs outputted by FMPL method. We can see that HC and AND graphs perform quite similarly, while the OR graph does a good job in the settings with less variables (where the true graph is relatively denser).

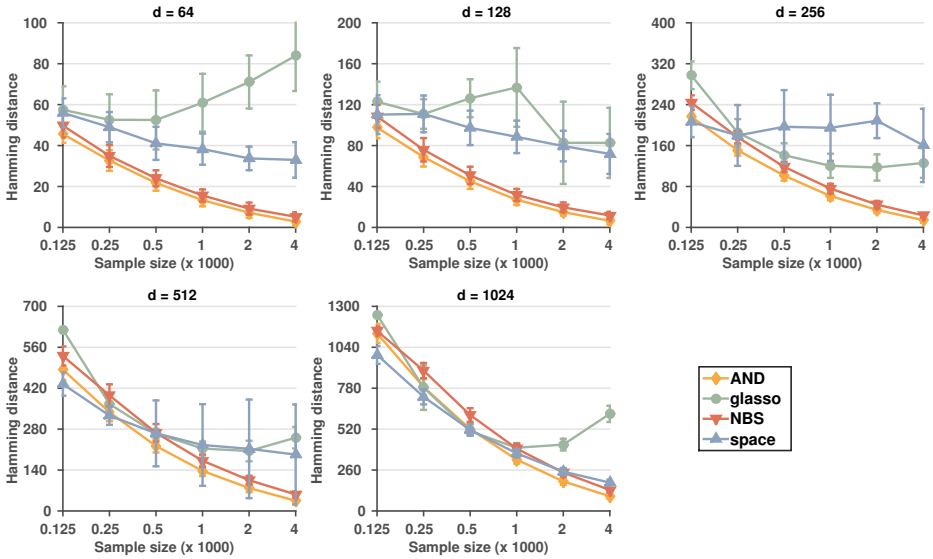

Figure 3.1: Hamming distance with different sample sizes in the structure learning experiments with synthetic data for FMPL and ℓ_1-methods. Here, AND graph represents the output of our proposed method. Figure reprinted from Article I.

When the Markov blanket searches were run on a standard 2.3 GHz workstation without utilizing parallelization, the running time of finding all the FMPL graphs ranged roughly from half a second, in case with $d = 64$ variables, to couple minutes in $d = 1024$ case.

3.2 Learning non-parametric graphical models

We continue under the theme of learning undirected graphical models. We will present a method for learning undirected graph structures without assuming any particular distribution the data should follow. In order to do this, we will first review some concepts from information theory and try to motivate why the Gaussian assumption might not always be suitable.

3.2.1 Going beyond Gaussian

In the last section our main underlying modeling assumption was that the data follows a multivariate normal distribution. This allowed us to derive a scoring function which was efficient to evaluate even for a high number

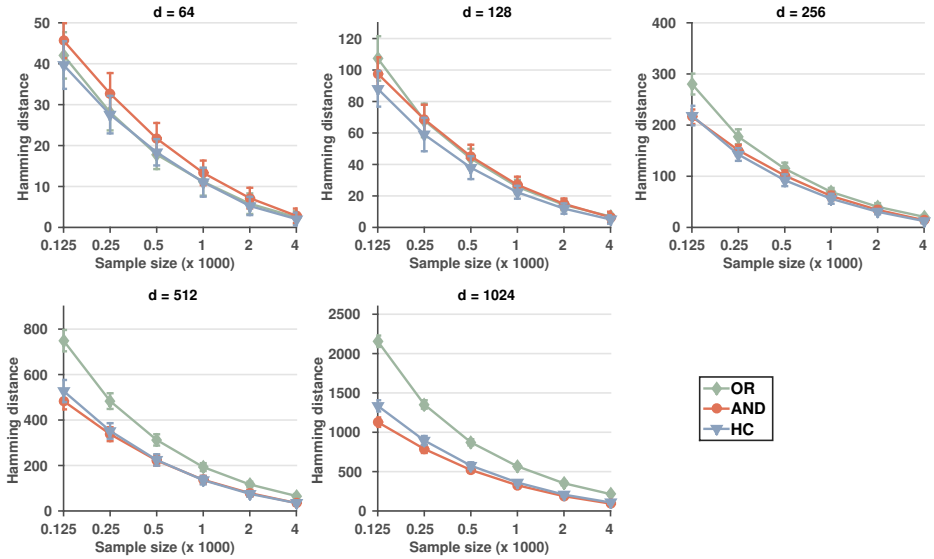

Figure 3.2: Hamming distance with different sample sizes in the structure learning experiments with synthetic data for different FMPL graphs. Figure reprinted from Article I.

of variables. However, assumption that our data follows a multivariate Gaussian distribution is quite rigid, and puts some constraints on the types of relationships between the variables we can model. To be more specific, by assuming normality, the task of deciding whether two variables are independent given some others reduces to testing whether the partial correlation between them is non-zero. As the correlation measures only the strength of a linear relationship, we might not to be able to detect dependencies correctly if the relationships are more complex than linear or if data deviates strongly from Gaussian distribution.

One class of methods that try to deal with this, while still retaining the Gaussian assumption partly, go under the name *Gaussian copulas* or *non-paranormal* methods [33, 34]. The main idea is to find univariate transformations for each variable so that after the transformation the data can be taken to follow multivariate normal distribution. The transformed data can be then dealt with using any method developed for the Gaussian data.

Kernel methods (see, for instance, [65]) are a second example of methods that are applicable in this situation. Their approach consists roughly of mapping the data to some possibly infinite dimensional space where the

representation of data allows one to detect complex relationships among the variables.

Our proposed solution to tackle with this problem will use mutual information to devise a non-parametric independence test which can be then paired with a constraint-based structure learning algorithm.

3.2.2 Mutual information and its estimation

Mutual information $I(X, Y)$ [9] measures the information that a random variable X carries from some other variable Y. For two continuous random variables with joint density $p_{XY}(x, y)$, it is defined as follows:

$$I(X,Y) = \int_{x\in\mathcal{X}} \int_{y\in\mathcal{Y}} p_{XY}(x,y) \log \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} dxdy, \tag{3.7}$$

where p_X and p_Y denote the marginal densities of the corresponding random variables. As a measure of association, mutual information is not restricted to detecting mere linear relationships like correlation. Mutual information equals zero if and only if the variables are independent.

However, estimating mutual information based on observed samples of X and Y might prove tricky in the general case, since if blindly following the definition, we would first need to estimate the densities appearing in (3.7). To bypass this, we will make use of the non-parametric mutual information estimator by Kraskov et al. [28] which relies on the k^{th}-nearest neighbor statistics computed from the observed data.

The Kraskov estimator is based on a previous entropy estimator from Kozachenko and Leonenko [27] which estimates the entropy using the assumption that the probability density is constant inside the hyperspheres containing the $k-1$ nearest neighbors of each data point. The resulting formula for entropy involves distances from each data point to their k^{th} nearest neighbors. As mutual information is expressible through (differential) entropies, denoted $H(\cdot)$, as

$$I(X,Y) = H(X) + H(Y) - H(X,Y), \tag{3.8}$$

Kraskov et al. apply the estimator to each of entropies appearing in Eq. (3.8) while taking into account that the length scales in joint and marginal spaces might be different, effectively canceling the aforementioned distance terms. After finding the k^{th} nearest neighbor of each point (x^i, y^i) in the joint space and recording the corresponding distance ϵ_i, the Kraskov mutual information estimator takes the form

$$\widehat{I}(X,Y) = \psi(n) + \psi(k) - \frac{1}{n}\sum_{i=1}^{n}[\psi(n_x(i)+1) + \psi(n_y(i)+1)], \tag{3.9}$$

where $\psi(\cdot)$ denotes the digamma function, n is the sample size and $n_x(i)$ stands for the number of points found around x^i in the marginal space of X within the distance ϵ_i (and n_y is defined similarly). The choice of the value for k is connected to bias-variance trade-off, as smaller k means that the assumption of constant density is made in smaller volumes.

As our ultimate interest will be to apply the estimator for structure learning, we need to measure association between random variables given the values of some other variables. To this end, we will need conditional mutual information $I(X, Y \mid Z)$. An important property regarding independence testing is that conditional mutual information is zero if the variables in question are conditionally independent, specifically

$$I(X, Y \mid Z) = 0 \text{ if and only if } X \perp\!\!\!\perp Y \mid Z.$$

Conditional mutual information also admits a decomposition through entropies which makes it possible to estimate it with similar techniques as used in ordinary Kraskov estimator. A formula resembling Eq. (3.9) for computing the conditional mutual information, $\widehat{I}(X, Y \mid Z)$, is provided by Vejmelka and Paluš [61]. Similarly to Eq. (3.9), computing the conditional mutual information requires performing the nearest neighbor search first in the joint space (X, Y, Z), and then counting points inside given radii in the marginal spaces.

3.2.3 Permutation test for conditional independence

While computing the value for conditional mutual information $\widehat{I}(X, Y \mid Z)$ proved to be straightforward, applying it to independence testing poses another challenge. Even if the random variables under consideration are conditionally independent, the estimator $\widehat{I}(X, Y \mid Z)$ generally never equals exactly zero due to random fluctuations.

If we knew the distribution of conditional mutual information estimator under the hypothesis of independence, we could just check where the observed value lands and use this to guide us when deciding on independence. However, to our best knowledge, the analytical form of the distribution of the estimator still remains elusive, and we need to resort to other means.

To cope with this problem, we will use a permutation test. Given the observed data of size n denoted $\boldsymbol{x} = (x^1, \ldots, x^n)$, $\boldsymbol{y} = (y^1, \ldots, y^n)$ and $\boldsymbol{z} = (z^1, \ldots, z^n)$, we try to simulate the conditional independence by randomly permuting the samples $\boldsymbol{y}$ and thus breaking the dependence between X and Y. The idea is to repeatedly permute the data T times, and compute mutual information $\widehat{I}(X, Y \mid Z)$ using the permuted data. This

gives us T new different values for mutual information. Using these, we can compare where the initially estimated value ranks among them. Denoting by K the number of permuted values that exceed the initial value, we get the following estimate for the p-value under the null hypothesis of independence:

$$\hat{p} = \frac{1+K}{1+T}.$$

This value can be then compared to predetermined significance level α in order to decide on the independence. This gives us our non-parametric test for conditional independence.

Regarding the computational complexity of mutual information based independence testing, although the permutation test can be done completely in parallel, the single conditional mutual information estimations require nearest neighbor searches that can get computationally demanding as the number of samples n, or the dimension (through adding more conditioning variables) gets large. The brute force approach would scale as $O(kdn^2)$, where d refers to the dimension of (X, Y, Z) space. Using data structures such as kd-trees [1] one can bring the complexity with respect to the sample size down to $O(n \log n)$.

In our implementation, we also included simple heuristic rules-of-thumb that determine couple situations when the permutation could be skipped and independence deduced. For instance, if fast-to-perform partial correlation based test accepts independence, and estimated conditional mutual information is below 0.001 nats, we accept the independence without performing the test. For more details, we refer to Sec. 2.3. in Article II.

Even though the permutation test described above breaks the dependence between $\boldsymbol{y}$ and $\boldsymbol{x}$ as desired, Runge [46] notes that this also results in the dependence between permuted $\boldsymbol{y}$ and conditioning $\boldsymbol{z}$ being broken which is in principle wrong when testing for conditional independence. He proposes a local permutation scheme to counter this. This approach involves defining neighborhoods for each data point in Z space, and then permuting $\boldsymbol{y}$ locally with help of these neighborhoods. This introduces an additional tunable hyperparameter, k_{perm}, defining the number of points in the neighborhood. We ran some experiments comparing local permutation scheme to the simple one and found out that local permutation strategy did not seem to result in a more accurate structure recovery when accompanied with the algorithm discussed in the next section. Therefore, we opt to use the simple permutation scheme. For details, we refer to Appendix A of Article II.

3.2.4 Empirical performance

To test the method in practice in structure learning, we need to accompany it with a constraint-based learning algorithm. Our strategy will resemble the first step of FMPL algorithm in the last section. That is, we use a constraint-based algorithm to learn the Markov blankets of each variable which we will then combine using the AND-rule, also described when discussing FMPL, to form the final undirected graph.

To learn the Markov blankets for each node, we use IAMB (incremental association Markov Blanket) algorithm [58]. The algorithm consists of two phases where in the first we add variables to the blanket if they are found to be conditionally dependent given the current blanket. Order in which variables are considered to be added is determined by a dynamic heuristic which in case of our non-parametric test will be the conditional mutual information. In the second phase, we try to identify the possible false positives by removing the nodes that are found to be conditionally independent given the remaining blanket.

The resulting method is referred to as knnMI. Next, we will briefly describe some of the results from experiments in Article II.

We will compare knnMI with $k = 5$ to other independence tests paired with the same structure learning algorithm. These tests include: Fisher-Z test for partial correlation (assumes Gaussian data, see, for instance, Kalisch and Bühlmann [22]), KCIT [65] which is a non-parametric kernel based method, and RCIT [53], also a kernel method but based on approximations in order to make the test more scalable. Significance level was set to $\alpha = 0.05$ for every test. In addition to these, we include also the familiar glasso and NBS methods which are applied to data after performing a non-paranormal transformation. These methods are referred to as NPN_glasso and NPN_mb, respectively.

In the experiments, our main interest was to study how the different methods react when the data generating mechanism deviates from the simple linear Gaussian structure. To that end, we created a data from a small network containing seven nodes and eight edges. To ease the data generation, the graph was selected to be chordal, meaning that we can represent it equally well using a DAG or an undirected graph. We used the DAG form, as it allows us to sample data for each variable given the values of its parents, and varied the following properties in the data generating mechanism:

1. Each variable X_i was defined either as a *linear* or a *non-linear* function of its parent variables plus an additive noise term, denoted ϵ_i. The

used non-linearities included, for instance, trigonometric functions, logarithm, and absolute value.

2. The additive noise distribution was taken to be either standard Gaussian, uniformly distributed between $[-1, 1]$, or standard t with two degrees of freedom.

Considering all the options for noise distributions in linear and non-linear settings yields six different scenarios. The results of applying aforementioned structure learning methods to these data sets with different sample sizes are presented in Figure 3.3. The goodness of the learned structures was measured using Hamming distance.

By looking at Figure 3.3, we can see that in the linear case (top-row), kernel methods and the Gaussian test are the most accurate. The proposed method knnMI seems to converge to the right graph but with a slower pace than the leading methods. However, we observe drastic difference in results when we change the relationships in the generating model to be non-linear. In this case (the bottom-row of Figure 3.3) knnMI is generally the best in capturing the underlying structure. KCIT is the runner-up while the other methods do not seem to be able to converge to the right structure.

We considered also a larger version of the non-linear data generating structure. The larger version was created by combining three seven node networks discussed above to create a network with 21 variables. The conclusion from this experiment was the same: knnMI and KCIT were the only methods that seemed to work consistently, knnMI slightly better with small sample sizes.

In addition to this, the methods were tested with non-paranormal data. The data was generated from a Gaussian undirected graphical model with 10 or 20 nodes, and then put through a power transformation. The non-paranormal methods were the best in the larger setting. The exact kernel method KCIT worked also well in these settings and the performance of knnMI was comparable to KCIT with the smallest and the largest sample sizes considered.

To summarize, our method was found to outperform the other methods in case the data generating mechanism involved non-linearities. Out of the tested methods, only the kernel method KCIT could achieve comparable performance in these settings. Comparison of computational complexities for these two methods favors knnMI as the KCIT scales cubicly in the sample size. That being said, all the other compared methods scale better to larger data sets. But as seen in the experiments, the gain in speed might come with a trade-off in accuracy.

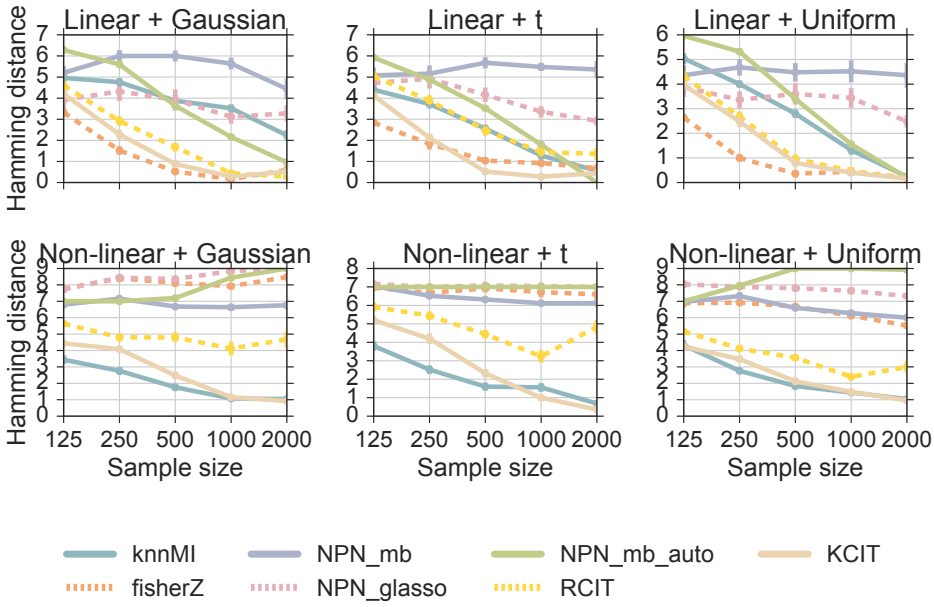

Figure 3.3: Structure learning with data generated using linear (top-row) and non-linear (bottom-row) relationships with additive noise (columns). Figure reprinted from Article II.

Chapter 4

Learning and applying directed graphical models

In this chapter, we change our setting in two major ways: 1) we consider discrete random variables instead of continuous, 2) the graphical models we study are directed, Bayesian networks. We start by going through some common scoring criteria that are used when learning Bayesian network structures, and then propose a new one, called Quotient Normalized Maximum Likelihood (qNML). After discussing qNML and its properties, we move on to an application of Bayesian networks. To that end, we review the concept of Fisher kernel and show how it can be combined with Bayesian networks to produce a similarity measure for categorical data vectors.

4.1 Scoring criteria for structure learning in the discrete setting

Let $\boldsymbol{X}$ denote a $n \times d$ data matrix consisting of n observations on d categorical variables $X = (X_1, \ldots, X_d)$. Recall from Chapter 2, that the Bayesian network consists of DAG G and parameters $\theta_{ijk} = P(X_i = k \mid X_{\pi(i)} = j)$. Recall also that Bayesian network allows us to express the likelihood function (see Eq. (2.4)) given n i.i.d. data points $\boldsymbol{X}$ as

$$p(\boldsymbol{X} \mid G, \theta) = \prod_{i=1}^{d} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}}, \tag{4.1}$$

where N_{ijk} denotes the number of times we observe X_i taking value k when its parents $X_{\pi(i)}$ take value j in our data $\boldsymbol{X}$.

All the scoring functions we consider next are decomposable. Recall that this means we can express them as a sum (or a product) of local scoring

functions: $f(G, \boldsymbol{X}) = \sum_i f_i(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)})$, where the local terms depend only on the data through variable X_i and its parents $X_{\pi(i)}$.

4.1.1 BDeu

As we have already seen in Chapter 3, one common choice for $f(G, \boldsymbol{X})$ is to consider the posterior probability of the graph given the observed data, $p(G \mid \boldsymbol{X})$, which simplifies to marginal likelihood, assuming uniform prior over different graphs.

In the discrete setting, assuming parameter sets θ_{ij} follow independent Dirichlet distributions, $\theta_{ij} \sim Dir(\alpha_{ij1}, \ldots, \alpha_{ijr_i})$, allows one to evaluate the marginal likelihood in closed form. The resulting score is called Bayesian-Dirichlet-score (BD) [18]. BD-score requires the user to specify hyperparameters α_{ijk} which might not be straightforward. A very widely used form of BD-score, called Bayesian-Diriclet equivalence uniform (BDeu), tries to solve this by requiring the user to provide only one tuning parameter: the equivalent sample size, $\alpha > 0$. Using this, the hyperparameters are set to $\alpha_{ijk} = \alpha/(r_i q_i)$. BDeu-score is also *score equivalent*. This property guarantees that two Bayesian networks G_1 and G_2 that imply exactly the same assertions of independence, are scored equally.

Even though depending only on one hyperparameter, the BDeu-score can be really sensitive with respect of this choice: in [50], it is shown how the highest scoring DAG structure can vary even from an empty graph to the full network only by changing the value of α.

Another shortcoming of BDeu was noted by Suzuki [54]. Suzuki shows how BDeu violates *regularity* in model selection. A decomposable scoring function $f_i(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)})$ is said not to be regular if it prefers a larger parent set over a smaller one, even though the larger set does not provide a better fit to data. The fit to data is here defined via empirical conditional entropy $H(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)})$, with the smaller entropy implying a better fit. The example in [54] where BDeu violates the regularity involves deterministic relationships between the variables. In [55], it is also argued that regular scores are more efficient when applied with branch-and-bound type search algorithms for structure learning.

4.1.2 BIC

Another very popular scoring function is Bayesian Information Criterion (BIC), which is derived as an asymptotic expansion of the log marginal

likelihood. For the Bayesian networks, it admits the following form:

$$BIC(G, \boldsymbol{X}) = \sum_{i=1}^{d} \log \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}) - \frac{k_i}{2} \log n, \tag{4.2}$$

where $\log \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}) = \max_{\theta_i} \log p(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}, \theta_i)$ denotes the maximized log-likelihood function for X_i given the parents $X_{\pi(i)}$, and $k_i = q_i(r_i - 1)$ is the number of free parameters in the conditional distribution of X_i.

BIC is often stated to have tendency to underfit, preferring simple models unless a lot of data is available. This behaviour is observed for instance in [51], where BIC requires large sample sizes before converging to the correct structure.

4.1.3 fNML

The factorized Normalized Maximum Likelihood (fNML) [51] is a close relative to the qNML criterion that we will present in the next section. Both of these criteria draw their motivation from information theory as they are approximations to a normalized maximum likelihood (NML) criterion which itself is an instance of a more general MDL principle.

Utilizing the NML for learning a Bayesian network structure requires us to compute the NML distribution under any given DAG G defined as

$$p_{NML}(\boldsymbol{X} \mid G) = \frac{p(\boldsymbol{X} \mid G, \hat{\theta}(\boldsymbol{X}))}{\sum_{\boldsymbol{X}' \in \mathcal{X}} p(\boldsymbol{X}' \mid G, \hat{\theta}(\boldsymbol{X}'))} = \frac{\prod_{i=1}^{d} \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)})}{\sum_{\boldsymbol{X}' \in \mathcal{X}} \prod_{i=1}^{d} \hat{p}(\boldsymbol{X}'_i \mid \boldsymbol{X}'_{\pi(i)})}, \tag{4.3}$$

where $\mathcal{X}$ represents the set off all possible $n \times d$ data matrices for our variables X. We have also used notation $\hat{\theta}(\cdot)$ to make the data set from which the maximum likelihood parameters are computed explicit. Taking a log of (4.3) and comparing the result to BIC formula (4.2) shows a clear resemblance, the only difference being the penalty term.

The logarithm of the penalty term, that is, the denominator in (4.3) is called *parametric complexity* or *regret*. This huge sum does not admit a simple factorization over variables, and for a general Bayesian network structure, this term is impossible to compute in reasonable time. However, in case of n observations on a single categorical variable X_i, the regret $reg(n, r_i) = \log \sum_{\boldsymbol{X}'_i \in \mathcal{X}_i} \hat{p}(\boldsymbol{X}'_i)$ and therefore the NML distribution, can be computed efficiently with an exact, linear time (with respect to n and r_i) algorithm [26] or by constant time approximations [56, 57]. Both, the fNML and the soon-to-be-presented qNML make use of these results.

The fNML solution for circumventing the intractability of the regret computation in (4.3) is to approximate regret by a similar in spirit, but decomposable object:

$$\log \sum_{\boldsymbol{X}' \in \mathcal{X}} p(\boldsymbol{X}' \mid G, \hat{\theta}(\boldsymbol{X}')) \approx \sum_{i=1}^{d} \log \sum_{\boldsymbol{X}'_i \in \mathcal{X}_i} \hat{p}(\boldsymbol{X}'_i \mid \boldsymbol{X}_{\pi(i)}). \tag{4.4}$$

While now being decomposable, the penalty term is also dependant on the observed data $\boldsymbol{X}$. After some manipulations [51], the local penalty (i^{th} term of the sum in (4.4)) can be expressed as

$$\log \sum_{\boldsymbol{X}'_i \in \mathcal{X}_i} \hat{p}(\boldsymbol{X}'_i \mid \boldsymbol{X}_{\pi(i)}) = \sum_{j=1}^{q_i} reg(N_{ij}, r_i), \tag{4.5}$$

where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ is the number of times we have observed the parents of X_i taking the value j. This gives us the fNML score:

$$\text{fNML}(G, \boldsymbol{X}) = \sum_{i=1}^{d} \log \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}) - \sum_{j=1}^{q_i} reg(N_{ij}, r_i). \tag{4.6}$$

When compared to BDeu, fNML does not require the user to set any hyperparameters which is an attractive property. Also empirically, the models learned by fNML are generally better in prediction than the ones given by BDeu [51]. However, fNML is not score equivalent. As seen in Article III, it seems also on some occasions to learn rather complex networks which would be hard to interpret.

4.2 qNML score

We are now ready to define the qNML score presented in Article III. The goal is to introduce a new score overcoming some of the negative theoretical properties of the previously mentioned criteria while also maintaining good empirical properties in terms of prediction.

4.2.1 Definition

Again, we aim at computing the NML distribution in (4.3), but resorting this time to a different approximation. First, consider the usual decomposition for the probability of $\boldsymbol{X}$ according to a DAG G

$$p(\boldsymbol{X} \mid G) = \prod_{i=1}^{d} p(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}) = \prod_{i=1}^{d} \frac{p(\boldsymbol{X}_{fa(i)})}{p(\boldsymbol{X}_{\pi(i)})}, \tag{4.7}$$

where the last equation is just the definition of conditional probability and $fa(i) = \pi(i) \cup \{i\}$.

Next, we will again utilize the fact that we know how to compute the NML distribution for a single categorical variable. To that end, we assume that the parent variables $X_{\pi(i)}$ form a fully connected graph. This means that we can treat $X_{\pi(i)}$ and $X_{fa(i)}$ as a two single categorical variables that can take q_i and $r_i \cdot q_i$ different values, respectively. Now, simply replacing the probabilities appearing in (4.7) by one-dimensional NML probabilities and taking the logarithm, we get the qNML score

$$\mathrm{qNML}(G, \boldsymbol{X}) = \sum_{i=1}^{d} \log \frac{p^1_{NML}(\boldsymbol{X}_{fa(i)})}{p^1_{NML}(\boldsymbol{X}_{\pi(i)})}, \tag{4.8}$$

where

$$\log p^1_{NML}(\boldsymbol{X}_A) = \log \hat{p}^1(\boldsymbol{X}_A) - reg(n, \prod_{i \in A} r_i) \tag{4.9}$$

with the superscripted 1 emphasizing that the multivariate input is treated as a single categorical variable.

By combining the maximized likelihood terms corresponding to variable sets $fa(i)$ and $\pi(i)$, qNML takes the familiar form of a penalized maximum likelihood:

$$\mathrm{qNML}(G, \boldsymbol{X}) = \sum_{i=1}^{d} \log \hat{p}(\boldsymbol{X}_i \mid \boldsymbol{X}_{\pi(i)}) - (reg(n, r_i \cdot q_i) - reg(n, q_i)) \tag{4.10}$$

Compared to fNML, an evident difference is that the penalty function is not dependent on the particular observed data vectors but only on the sample size n and the numbers of possible categories for the variables. Similarly to BIC and fNML, qNML is completely free of any tunable hyperparameters.

4.2.2 Theoretical properties of the score

We will now review the properties of qNML. For the proofs, reader is referred to Article III and its Supplementary Material.

All the scoring functions mentioned in the previous section are consistent, meaning roughly that they will give the highest score to the data generating network when enough data is available. qNML is not an exception to this:

Theorem 4.1 (Sec. 3.2 in Article III)**.** *qNML is consistent.*

When discussing fNML, we noted that it is not score equivalent. One can, in a quite straightforward way, verify the following for qNML:

Theorem 4.2 (Sec 3.1 in Article III)**.** *qNML is score equivalent.*

Regarding BDeu, we noted the recent finding of it not being a regular scoring function. After some slightly tedious derivations, one can show that:

Theorem 4.3 (Appendix B of Article III)**.** *qNML is regular.*

As qNML and fNML are approximations to NML, it is interesting to ask if there exists situations where they would be equal to the exact NML. For the fNML, the equality holds only in the case when the score is evaluated for an empty network [45]. qNML too agrees with NML on empty networks but also in cases where the network consists of separate, fully connected subgraphs.

4.2.3 On the empirical performance

The properties mentioned in the last section demonstrate that the qNML holds certain theoretical advantages over BDeu and fNML but we would naturally want this to translate into a good empirical performance. To that end, Article III features experiments where qNML was compared to the three previously mentioned scoring criteria in structure learning and prediction.

In the structure learning experiments, data sets of sizes ranging from 10 to 10^3 were created from seven known ground truth network structures. The highest scoring structures for each criterion were found using an exact structure learning algorithm and the discrepancy to the ground truth was measured using Structural Hamming Distance (SHD) [59]. The SHD measures similarity by counting the missing, extra or wrongly oriented edges as compared to the true network, while taking into account that not all the edge directions are distinguishable statistically.

We ranked the methods from 1 (the best) to 4 (the worst) according to the SHD in each performed experiment. Looking at the average rank over all the networks and repeated experiments at different sample sizes, qNML was found never to have the worst average ranking. It had the best ranking for sample sizes greater than 300. On the smaller sample sizes BIC was the best but it did not fare so well with the larger ones.

In the prediction experiments, the scoring criteria were compared using 20 data sets from UCI repository. The data sets were split in training and test sets with varying proportions. The network structure and the parameters were learned from the training data and the performance was measured by recording the negative log-likelihood evaluated for the test set. Ranking the scores again from best to worst according to predictive

performance and then averaging over all the data sets, sample sizes and experiment repetitions, resulted fNML to have the best ranking with qNML as the runner-up. BIC seemed the best with the smallest data sets and the worst with the biggest ones. fNML fared the best with the bigger data sets and the performance of qNML was usually somewhere between them, making it a safe choice across all the sample sizes.

Lastly, a closer look was taken into the average number of parameters in the learned networks with the same data sets and 10% of data used for training. The previously mentioned observation that fNML seemed to sometimes prefer quite complex networks was visible in the results (7/20 cases the network chosen by fNML was the most complex one) and qNML was observed to produce sparser networks on average.

To conclude, none of the scores could be declared as winner in all situations but qNML seemed to offer an overall robust and safe choice, being generally never the worst one in any of the tasks or settings considered.

4.3 Application: Bayesian network Fisher kernel

Until this point, we have been mostly dedicated to different strategies for learning the network structures from observational data. Next, we move on to a different part of the pipeline and present an application where we make use of the fully specified DAG model which can be learned, for instance, by plugging any of the scores from the previous section in to some search algorithm. This section is based on Article IV.

Our main goal here is to form a similarity measure for d-dimensional categorical data vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. We assume that $\boldsymbol{x}$ and $\boldsymbol{y}$ are realizations of $X = (X_1, \ldots, X_d)$ whose distribution we model using a Bayesian network model $M = (G, \theta)$, where G denotes the DAG and θ the parameters.

In general, measuring similarity between categorical vectors is not as straightforward as measuring similarity of quantitative data vectors. Nevertheless, various similarity measures appear in literature [5], the simplest ones basically count the number of components where the vectors agree. Literature on similarity measures that would take into account the possible dependencies between the components is scarce. Some approaches exist that consider the pairwise dependencies between components [11, 35, 39, 43]. A Bayesian network would provide a natural solution to this by modeling the whole dependency structure, beyond the pairwise one, with the help of a DAG.

In order to make use of the Bayesian network, we will build a Fisher kernel based on it. Fisher kernels have been constructed for some models

that can be expressed as Bayesian networks (Hidden Markov Models [21] and Probabilistic Latent Semantic Analysis [7]) but the general formula for the Fisher kernel under any DAG model over categorical variables seems to be missing from the literature.

4.3.1 Fisher kernel

Kernel methods [49] are based on embedding the observed data vectors to some other (possible infinite dimensional) space called the feature space. The kernel function, denoted $K(\boldsymbol{x}, \boldsymbol{y})$, allows us to evaluate the inner products for observed data vectors in this feature space without ever explicitly constructing the feature representations. As a myriad of machine learning algorithms can be written in terms of inner products between vectors, kernel methods provide us a convenient way for discovering relationships between the data vectors in complex feature spaces.

Fisher kernel [20] provides a way of constructing a kernel function with the aid of a parametric probabilistic model $p(\cdot \mid \theta)$. Assume that θ contains k free parameters. Fisher kernel is defined as follows:

$$K(\boldsymbol{x}, \boldsymbol{y}; \theta) = \nabla_\theta \log p(\boldsymbol{x} \mid \theta)^T \mathcal{I}^{-1}(\theta) \nabla_\theta \log p(\boldsymbol{y} \mid \theta), \tag{4.11}$$

where $\nabla_\theta \log p(\boldsymbol{x} \mid \theta) \in \mathbb{R}^k$ is the gradient vector (wrt. parameters) evaluated at $\boldsymbol{x}$ and $\mathcal{I}^{-1}(\theta)$ denotes the inverse of the Fisher information matrix. Fisher information is a $k \times k$ matrix with elements defined as

$$I(\theta)_{ij} = -\mathbb{E}_X \left[\frac{\partial^2 \log p(X \mid \theta')}{\partial \theta_i' \theta_j'} \right]_{\theta'=\theta}. \tag{4.12}$$

Intuitively, Fisher kernel measures similarity by looking at in which direction $\boldsymbol{x}$ would like to change the parameters of the model which is quantified via the gradient vector $\nabla_\theta \log p(\boldsymbol{x} \mid \theta)$. If the gradient for the $\boldsymbol{y}$ points to the same direction as $\nabla_\theta \log p(\boldsymbol{x} \mid \theta)$, then the two points would be deemed similar.

4.3.2 Fisher kernel for Bayesian networks

In order to derive the Fisher kernel for Bayesian networks, we just need to replace the likelihood function $p(\cdot \mid \theta)$ appearing in (4.11) by the likelihood function given by some Bayesian network model $M = (G, \theta)$ (see, Eq. (4.1)). This results in a straightforward, albeit a bit lengthy and messy calculation,

which gives us the following result:

$$K(\boldsymbol{x}, \boldsymbol{y}; \theta) = \sum_{i=1}^{d} K_i(\boldsymbol{x}, \boldsymbol{y}; \theta), \text{ where} \tag{4.13}$$

$$K_i(\boldsymbol{x}, \boldsymbol{y}; \theta) = \begin{cases} 0, & \text{if } \boldsymbol{x}_{\pi(i)} \neq \boldsymbol{y}_{\pi(i)}, \\ -\frac{1}{p(\boldsymbol{x}_{\pi(i)}|\theta)}, & \text{if } \boldsymbol{x}_{\pi(i)} = \boldsymbol{y}_{\pi(i)} \\ & \text{and } x_i \neq y_i, \\ \frac{1}{p(\boldsymbol{x}_{\pi(i)}|\theta)} \cdot \left(\frac{1}{\theta_{i\boldsymbol{x}_{\pi(i)} x_i}} - 1 \right), & \text{if } \boldsymbol{x}_{\pi(i)} = \boldsymbol{y}_{\pi(i)} \\ & \text{and } x_i = y_i. \end{cases}$$

For the detailed derivation, we refer to the Article IV.

To put (4.13) into words, Fisher kernel for Bayesian networks decomposes into a sum over nodes i. Looking at the local term $K_i(\boldsymbol{x}, \boldsymbol{y}; \theta)$, we see that this term evaluates to zero, if the parent variables $X_{\pi(i)}$ take different values in data vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. If the values for parents agree, the value for this term depends on the values of X_i in $\boldsymbol{x}$ and $\boldsymbol{y}$. In case they do not agree, $x_i \neq y_i$, the local term takes a negative value: $-1/p(\boldsymbol{x}_{\pi(i)} \mid \theta)$, which involves the marginal probability of the parent variables. And in case they agree, the value for the local term is $1/p(\boldsymbol{x}_{\pi(i)} \mid \theta)(1/\theta_{i\boldsymbol{x}_{\pi(i)} x_i} - 1)$, in which $\theta_{i\boldsymbol{x}_{\pi(i)} x_i} = p(X_i = x_i \mid X_{\pi(i)} = \boldsymbol{x}_{\pi(i)})$.

4.3.3 Properties of the kernel

As we have noted couple times before, some Bayesian network structures are equivalent in a sense that even though not being structurally exactly similar, they still represent the same assertions of conditional independence, and can be used to represent the same set of joint distributions with the right choice of parameters. This gives rise to a question whether the resulting Fisher kernel is dependent on the used network structure. The answer is given in the following theorem:

Theorem 4.4 (Theorem 3 in Article IV)**.** *Let $M_1 = (G_1, \theta^1)$ and $M_2 = (G_2, \theta^2)$ be two Bayesian networks with equivalent structures such that they represent the same distribution over X. Now,*

$$K(\boldsymbol{x}, \boldsymbol{y}; \theta^1) = K(\boldsymbol{x}, \boldsymbol{y}; \theta^2), \ \forall \boldsymbol{x}, \boldsymbol{y}.$$

In other words, Bayesian network Fisher kernel is indifferent to whichever of the equivalent structures we use to construct it.

It is also easy to generalize the Fisher kernel to comparing pairs of i.i.d. observation sets instead of only pairs of observations. Namely, the Fisher kernel between two sets of i.i.d. observations is the average over all the pairwise kernel evaluations for these two sets [7]. To express this formally, let $\boldsymbol{X} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n)$ and $\boldsymbol{Y} = (\boldsymbol{y}^1, \ldots, \boldsymbol{y}^m)$ denote two sets of observations of size n and m, respectively. Now, the set kernel is defined as:

$$K_S(\boldsymbol{X}, \boldsymbol{Y}; \theta) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} K(\boldsymbol{x}^i, \boldsymbol{y}^j; \theta), \tag{4.14}$$

where the right-hand side could be further simplified by expressing it with help of the counts of different variable and parent combinations in data sets $\boldsymbol{X}$ and $\boldsymbol{Y}$.

Since the set kernel in (4.14) allows us to compute inner products between sets of observations, we can use this in order to compute distances between these sets too. This follows by applying the kernel trick [49] to Euclidean distance, that is, expressing the usual (squared) Euclidean distance with the help of the inner products between vectors, and replacing those with the set kernel evaluations:

$$d(\boldsymbol{X}, \boldsymbol{Y}) = K_S(\boldsymbol{X}, \boldsymbol{X}; \theta) + K_S(\boldsymbol{Y}, \boldsymbol{Y}; \theta) - 2 \cdot K_S(\boldsymbol{X}, \boldsymbol{Y}; \theta). \tag{4.15}$$

In Article IV we also point out the connection between the distance given by (4.15) and the empirical version of a kernel based statistic called Maximum Mean Discrepancy (MMD) [16] which can be used for testing whether two data sets originate from a common distribution.

In the next section, we discuss experiments where we make use of $d(\boldsymbol{X}, \boldsymbol{Y})$ and refer to it as MMD distance. Even though calling the distance MMD, we note that in order to make use of all the machinery and theoretical results regarding MMD developed in [16] would require checking certain theoretical properties of the Fisher kernel, which we do not pursue here.

4.3.4 Applying the Fisher kernel

To demonstrate the nature of the similarity that Bayesian network Fisher kernel measures, we drew inspiration from the area of interpretable machine learning (see, for instance, [37]). Our goal is to gain insight into the underlying Bayesian network model upon which the kernel is built by using the kernel to find data points that are representative for the model. As noted in [23], Fisher kernel is naturally suited to these kinds of interpretation tasks as the similarity for the data points is defined from the point of the view of the model.

Our experiments are similar in spirit to the data summarization experiments in [23]. In the experiments, we first learn a Bayesian network model based on a large training data set $\boldsymbol{X}$. Then we construct a Fisher kernel based on the network and use it to seek a much smaller representative subset of observations $\boldsymbol{X}^S$ of size $k \equiv |S|$ that are "important" according to the model. By important we roughly mean that the performance of the model whose parameters are re-trained using only $\boldsymbol{X}^S$ should not deteriorate too much in predictive tasks when compared to the original model based on the full data $\boldsymbol{X}$. Here $S \subset \{1, \ldots, n\}$ denotes the indices of the data points that are included in this subset.

In practice, the representative subset is found by using a simple greedy algorithm aiming to minimize the MMD $d(\boldsymbol{X}^S, \boldsymbol{X})$ with respect to S. Details on the algorithm can be found in the Article IV.

The algorithm was applied to three data sets from the UCI repository: letter ($n = 20000, d = 17$), nursery ($n = 12960, d = 9$) and waveform-5000 ($n = 5000, d = 41$). The data sets were split in half to form training and test sets. The Bayesian network model was learned from training set using a hill-climbing algorithm with BIC as the scoring function. In order to quantify the goodness of the found representative subsets, we first recorded the log-likelihood for the test set which was obtained using the model trained on the full training set. Then the representative subsets with varying size of k (between 200 and 1000) were found using the algorithm. With each of the found subsets, the parameters of the model were retrained using the subset $\boldsymbol{X}^S$ and the resulting model was used in computing the test set log-likelihood. The results are shown in Figure 4.1.

In Figure 4.1, Fisher kernel based subset selection (greedy Fisher MMD) is compared to a baseline (random baseline), where the network parameters are retrained based on a randomly sampled subset $\boldsymbol{X}^S$. Each randomly trained network is used in computing the test set log-likelihood and the average result over 1000 subsets is shown in the figure.

In addition to a random baseline, we included a heuristic method (greedy ChiSq heuristic) which tries to find subset $\boldsymbol{X}^S$ in which all the marginal distributions of variables match well with the marginals in the whole set $\boldsymbol{X}$. Criterion is based on the classical χ^2-test and it is paired with the same greedy optimization algorithm as the Fisher kernel MMD.

Fisher kernel based MMD was also applied to Bayesian network classifiers. We used Tree Augmented Naive Bayes (TAN) [13] and K-dependence Bayesian (KDB) [48] classifiers. In the classification experiments, the performance was evaluated with (conditional) log-likelihood and misclassification rate on the test set. Otherwise the test setting was exactly as described

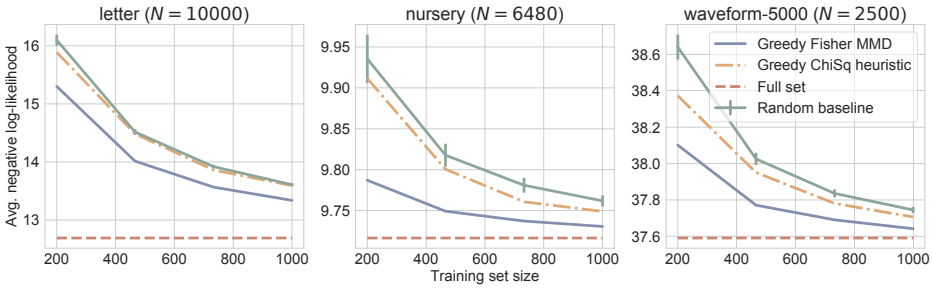

Figure 4.1: Negative log-likelihoods for different subset sizes and criteria in UCI datasets. Full set represents the accuracy of the model trained with all the available data and random baseline is averaged result over models retrained with random subsets. Figure adapted from Article IV.

previously. We show results for data sets letter and nursery for TAN and KDB-3 in Figures 4.2 and 4.3, respectively. The results in waveform-5000 were similar.

Looking at the results, we can see that Fisher kernel based subset selection criterion seems to generally find subsets that are important, capturing well the main characteristics in the whole data set, and thus resulting in a predictive performance that is closer to the baseline than the compared methods. This is especially seen in Figures 4.1 and 4.2. However, we note that in the classification experiments with the smallest sample sizes, KDB classifier with $K = 3$ did not always have better performance than the compared methods. This can be seen in Figure 4.3 with the data set nursery. One thing explaining this might be that Fisher kernel measures the similarity of data points with respect the whole model through all of its parameters, whereas the evaluation criteria in the classification experiment are strongly focused on the class variable and parameters governing its conditional distribution.

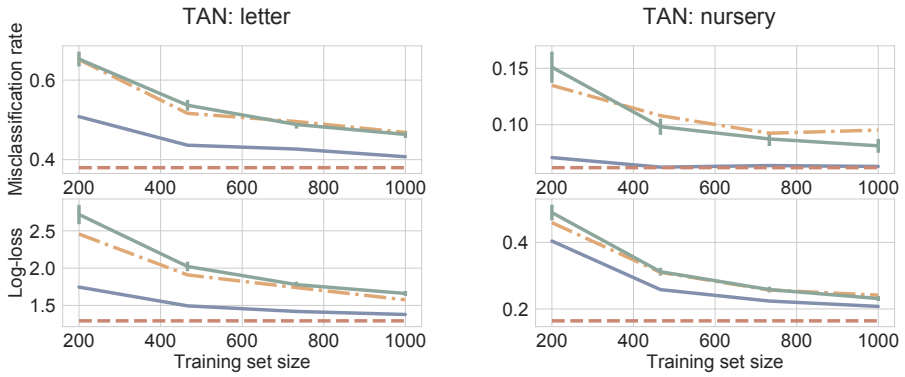

Figure 4.2: Results on classification experiments for TAN in data sets letter (left) and nursery (right). Missclassification rate on top and log-loss on the bottom. Labeling of the lines follows Figure 4.1. Figure adapted from Article IV.

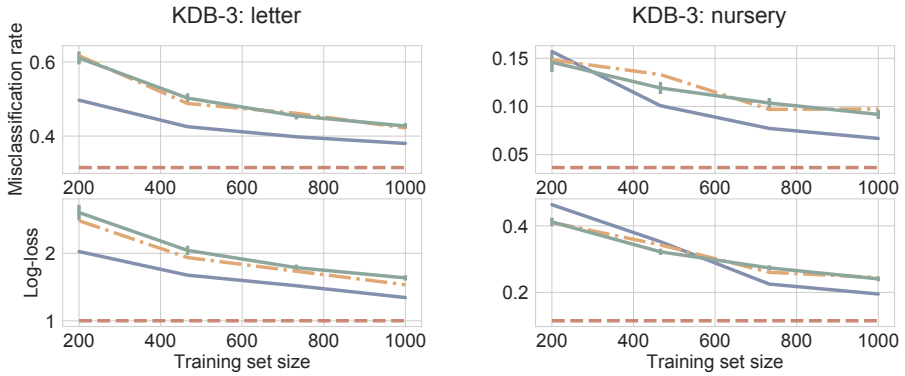

Figure 4.3: Results on classification experiments for KDB ($K = 3$) in data sets letter (left) and nursery (right). Missclassification rate on top and log-loss on the bottom. Labeling of the lines follows Figure 4.1. Figure adapted from Article IV.

Chapter 5

Conclusions

In this thesis, we have discussed graphical models, mainly focusing on maybe the most challenging task related to them: learning the graph structure from observed data. The discussion about the original research articles forming up this thesis was divided in two chapters treating undirected and directed graphical models.

The chapter on learning the undirected graphical models introduced to us two different methods for learning the graph structures. The profound difference of methods lies in their way of modeling the distribution of the input data. First, we presented the FMPL method which relied on somewhat rigid assumption on the multivariate normally distributed data. However, this assumption with the aid of the technique called pseudo-likelihood allowed us to derive a provably consistent and tuning-parameter free scoring function that was demonstrated in the experiments to be applicable even in high-dimensional settings.

Next, letting go of the Gaussian assumption, we used mutual information and permutation test to devise a non-parametric conditional independence test. In the numerical experiments, we found the method to work really well when the data generating mechanism involved non-linearities.

After this, we moved on to the realm of directed graphical models. We introduced a new scoring criterion called qNML, studied its theoretical properties and conducted numerical experiments. Even though none of the scores dominated in all aspects of the experimental evaluation, qNML maintained robust performance and was concluded to be an overall safe choice across a wide variety of settings.

Finally, we discussed how Bayesian networks can be used in defining a similarity measure over categorical feature vectors with help of an Fisher kernel. We derived a closed form formula for this Bayesian network Fisher kernel, and applied it to a task of finding small representative sets of

samples with respect to the model from larger training sets, a task drawing its motivation from the literature on interpretable machine learning.

We can see that a recurring theme in all the problems we have discussed in this thesis is some intractable, completely unknown, or hard-to-define quantity relevant to our learning problem: with Article I it is the marginal likelihood which plays a similar role as the normalized maximum likelihood discussed in Article III. Both of these we know how to compute, but the problem becomes intractable beyond toy examples. In Article II, the quantity is the sampling distribution of mutual information estimator, not a single number but something that we do not know how to express analytically. Finally, in Article IV, this quantity is the similarity between multivariate categorical data which does not even admit a single, clear definition.

Having proposed some strategies to overcome the aforementioned problems, we can still think of future directions that might provide interesting problems to consider in the follow-up work:

1. Regarding the mutual information estimation, analytic form for the sampling distribution would naturally be an important and ambitious goal. A more straightforward route to make the estimation faster would be to consider some approximate nearest neighbor search schemes. But an important question here is how to keep the accuracy not deteriorating too much and at the same time obtain worthwhile computational savings.

2. In Article III we noted that qNML agrees with the exact NML for specific networks structures. But what is the relation between these two scores for other networks? As the two criteria differ only on the penalty term, the difference is not dependent on the specific observed data, only on the sample size and the number of categories of variables.

3. Regarding Fisher kernel, for discrete Bayesian networks, it is possible that the conditional distribution of X given its parents Z is exactly the same for two values $Z = z_1$ and $Z = z_2$. Intuitively, one would then think that the vectors (x, z_1) and (x, z_2) should be similar. However, the contribution of X to the total Fisher similarity between these example vectors would be zero, as the parents do not match, $z_1 \neq z_2$. One way to address this could be to consider a model that introduces some structure to the local distributions by using only a single parameter vector for modeling both $p(X \mid Z = z_1)$ and $p(X \mid Z = z_2)$. This would naturally change the form of log-likelihood, and thus require us to derive Fisher kernel again at least partly.

References

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[2] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley & Sons, 1994.

[3] J. E. Besag. Nearest-neighbour systems and the auto-logistic model for binary data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(1):75–83, 1972.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.

[5] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In *SIAM Data Mining Conference*, pages 243–254, 2008.

[6] G. Casella and R. L. Berger. *Statistical Inference*. Wadsworth Group, Duxbury, 2002.

[7] J.-C. Chappelier and E. Eckard. PLSI: The true Fisher kernel and beyond. In *Machine Learning and Knowledge Discovery in Databases*, pages 195–210, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[8] G. Consonni and L. La Rocca. Objective Bayes factors for Gaussian directed acyclic graphical models. *Scandinavian Journal of Statistics*, 39(4):743–756, 2012.

[9] T. M. Cover and J. A. Thomas. *Elements of Information Theory 2nd Edition*. Wiley-Interscience, 2006.

[10] A. P. Dawid and S. L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *The Annals of Statistics*, 21(3):1272–1317, 1993.

[11] A. Desai, H. Singh, and V. Pudi. DISC: Data-intensive similarity measure for categorical data. In *Advances in Knowledge Discovery and Data Mining: 15th Pacific-Asia Conference*, pages 469–481. Springer, Berlin, Heidelberg, 2011.

[12] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[13] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2):131–163, 1997.

[14] D. Geiger and D. Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *The Annals of Statistics*, 30(5):1412–1440, 2002.

[15] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2014.

[16] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *JMLR*, 13:723–773, 2012.

[17] P. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

[18] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, Sep 1995.

[19] J. S. Ide, S. Zhang, and C. R. Li. Bayesian network models in brain functional connectivity analysis. *International Journal of Approximate Reasoning*, 55(1, Part 1):23 – 35, 2014. Applications of Bayesian Networks.

[20] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*, pages 487–493. MIT Press, 1998.

[21] T. S. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *Journal of computational biology*, 7(1-2):95–114, 2000.

[22] M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *JMLR*, 8:613–636, 2007.

[23] R. Khanna, B. Kim, J. Ghosh, and S. Koyejo. Interpreting black box predictions using Fisher kernels. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 3382–3390. PMLR, 16–18 Apr 2019.

[24] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[25] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

[26] P. Kontkanen and P. Myllymäki. A linear-time algorithm for computing the multinomial stochastic complexity. *Information Processing Letters*, 103(6):227–233, 2007.

[27] L. Kozachenko and N. N. Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

[28] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical Review E*, 69(6), 2004.

[29] S. Lauritzen. *Graphical Models*. Clarendon Press, 1996.

[30] J. Leppä-aho, J. Pensar, T. Roos, and J. Corander. Learning Gaussian graphical models with fractional marginal pseudo-likelihood. *International Journal of Approximate Reasoning*, 83:21 – 42, 2017.

[31] J. Leppä-aho, S. Räisänen, X. Yang, and T. Roos. Learning non-parametric Markov networks with mutual information. In V. Kratochvíl and M. Studený, editors, *Proceedings of the Ninth International Conference on Probabilistic Graphical Models*, volume 72 of *Proceedings of Machine Learning Research*, pages 213–224, Prague, Czech Republic, 11–14 Sep 2018. PMLR.

[32] J. Leppä-aho, T. Silander, and T. Roos. Bayesian network Fisher kernel for categorical feature spaces. Accepted for publication in *Behaviormetrika*, 2019.

[33] H. Liu, J. Lafferty, and L. Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *JMLR*, 10: 2295–2328, 2009.

[34] H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman. High-dimensional semiparametric Gaussian copula graphical models. *The Annals of Statistics*, 40(4):2293–2326, 2012.

[35] B. McCane and M. Albert. Distance functions for categorical and mixed variables. *Pattern Recognition Letters*, 29(7):986–993, 2008.

[36] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34(3):1436–1462, 2006.

[37] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu. Interpretable machine learning: Definitions, methods, and applications. *arXiv e-prints*, art. arXiv:1901.04592, Jan 2019.

[38] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[39] H. Niitsuma and T. Okada. Covariance and PCA for categorical variables. In T. B. Ho, D. Cheung, and H. Liu, editors, *Advances in Knowledge Discovery and Data Mining: 9th Pacific-Asia Conference*, pages 523–528, Hanoi, Vietnam, May 18–20 2005. Springer, Berlin, Heidelberg.

[40] A. O'Hagan. Fractional Bayes factors for model comparison. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):99–138, 1995.

[41] J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

[42] J. Pensar, H. Nyman, J. Niiranen, and J. Corander. Marginal pseudo-likelihood learning of discrete Markov network structures. *Bayesian Analysis*, 12(4):1195–1215, 2017.

[43] M. Ring, F. Otto, M. Becker, T. Niebler, D. Landes, and A. Hotho. ConDist: A context-driven categorical distance measure. In A. Appice, P. P. Rodrigues, V. Santos Costa, C. Soares, J. Gama, and A. Jorge, editors, *Machine Learning and Knowledge Discovery in Databases: European Conference*, pages 251–266, Porto, Portugal, September 7–11 2015. Springer International Publishing.

[44] J. Rissanen. Modeling by shortest data description. *Automatica*, 14: 445–471, 1978.

[45] T. Roos, T. Silander, P. Kontkanen, and M. P. Bayesian network structure learning using factorized NML universal models. In *Proceedings*

of the Information Theory and Applications Workshop (ITA-08), San Diego, CA, January 2008.

[46] J. Runge. Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pages 938–947, 2018.

[47] K. Sachs, D. Gifford, T. Jaakkola, P. Sorger, and D. A. Lauffenburger. Bayesian network approach to cell signaling pathway modeling. *Science's STKE*, 2002(148):pe38, 2002.

[48] M. Sahami. Learning limited dependence Bayesian classifiers. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 335–338. AAAI Press, 1996.

[49] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[50] T. Silander, P. Kontkanen, and P. Myllymäki. On sensitivity of the MAP Bayesian network structure to the equivalent sample size parameter. In R. Parr and L. van der Gaag, editors, *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI–07)*, pages 360–367. AUAI Press, 2007.

[51] T. Silander, T. Roos, and P. Myllymäki. Learning locally minimax optimal Bayesian networks. *International Journal of Approximate Reasoning*, 51(5):544 – 557, 2010.

[52] T. Silander, J. Leppä-aho, E. Jääsaari, and T. Roos. Quotient normalized maximum likelihood criterion for learning Bayesian network structures. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 948–957, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

[53] E. V. Strobl, K. Zhang, and S. Visweswaran. Approximate kernel-based conditional independence tests for fast non-parametric causal discovery. *ArXiv e-prints*, 2017.

[54] J. Suzuki. A theoretical analysis of the BDeu scores in Bayesian network structure learning. *Behaviormetrika*, 44(1):97–116, 2017.

[55] J. Suzuki and J. Kawahara. Branch and bound for regular Bayesian network structure learning. The 33rd Conference on Uncertainty in Artificial Intelligence, UAI 2017, August 11-15, 2017, Sydney, Australia, 2017.

[56] W. Szpankowski. *Average case analysis of algorithms on sequences*. John Wiley & Sons, 2001.

[57] W. Szpankowski and M. J. Weinberger. Minimax pointwise redundancy for memoryless models over large alphabets. *IEEE Transactions on Information Theory*, 58(7):4094–4104, July 2012.

[58] I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Algorithms for large scale Markov blanket discovery. In I. Russell and S. Haller, editors, *The 16th International FLAIRS Conference*, pages 376–380. AAAI Press, 2003.

[59] I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

[60] C. Uhler, A. Lenkoski, and D. Richards. Exact formulas for the normalizing constants of Wishart distributions for graphical models. *The Annals of Statistics*, 46(1):90–118, 2018.

[61] M. Vejmelka and M. Paluš. Inferring the directionality of coupling with conditional mutual information. *Physical Review E*, 77, 2008.

[62] H. Wang and S. Z. Li. Efficient Gaussian graphical model determination under G-Wishart prior distributions. *Electronic Journal of Statistics*, 6:168–198, 2012.

[63] J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons, 1990.

[64] D. M. Witten, J. H. Friedman, and N. Simon. New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20(4):892–900, 2011.

[65] K. Zhang, J. Peters, D. Janzing, and B. Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 804–813, 2011.

TIETOJENKÄSITTELYTIETEEN OSASTO
PL 68 (Pietari Kalmin katu 5)
00014 Helsingin yliopisto

DEPARTMENT OF COMPUTER SCIENCE
P.O. Box 68 (Pietari Kalmin katu 5)
FI-00014 University of Helsinki, FINLAND

JULKAISUSARJA **A**

SERIES OF PUBLICATIONS **A**

Reports are available on the e-thesis site of the University of Helsinki.

A-2014-1 J. Korhonen: Graph and Hypergraph Decompositions for Exact Algorithms. 62+66 pp. (Ph.D. Thesis)

A-2014-2 J. Paalasmaa: Monitoring Sleep with Force Sensor Measurement. 59+47 pp. (Ph.D. Thesis)

A-2014-3 L. Langohr: Methods for Finding Interesting Nodes in Weighted Graphs. 70+54 pp. (Ph.D. Thesis)

A-2014-4 S. Bhattacharya: Continuous Context Inference on Mobile Platforms. 94+67 pp. (Ph.D. Thesis)

A-2014-5 E. Lagerspetz: Collaborative Mobile Energy Awareness. 60+46 pp. (Ph.D. Thesis)

A-2015-1 L. Wang: Content, Topology and Cooperation in In-network Caching. 190 pp. (Ph.D. Thesis)

A-2015-2 T. Niinimäki: Approximation Strategies for Structure Learning in Bayesian Networks. 64+93 pp. (Ph.D. Thesis)

A-2015-3 D. Kempa: Efficient Construction of Fundamental Data Structures in Large-Scale Text Indexing. 68+88 pp. (Ph.D. Thesis)

A-2015-4 K. Zhao: Understanding Urban Human Mobility for Network Applications. 62+46 pp. (Ph.D. Thesis)

A-2015-5 A. Laaksonen: Algorithms for Melody Search and Transcription. 36+54 pp. (Ph.D. Thesis)

A-2015-6 Y. Ding: Collaborative Traffic Offloading for Mobile Systems. 223 pp. (Ph.D. Thesis)

A-2015-7 F. Fagerholm: Software Developer Experience: Case Studies in Lean-Agile and Open Source Environments. 118+68 pp. (Ph.D. Thesis)

A-2016-1 T. Ahonen: Cover Song Identification using Compression-based Distance Measures. 122+25 pp. (Ph.D. Thesis)

A-2016-2 O. Gross: World Associations as a Language Model for Generative and Creative Tasks. 60+10+54 pp. (Ph.D. Thesis)

A-2016-3 J. Määttä: Model Selection Methods for Linear Regression and Phylogenetic Reconstruction. 44+73 pp. (Ph.D. Thesis)

A-2016-4 J. Toivanen: Methods and Models in Linguistic and Musical Computational Creativity. 56+8+79 pp. (Ph.D. Thesis)

A-2016-5 K. Athukorala: Information Search as Adaptive Interaction. 122 pp. (Ph.D. Thesis)

A-2016-6 J.-K. Kangas: Combinatorial Algorithms with Applications in Learning Graphical Models. 66+90 pp. (Ph.D. Thesis)

A-2017-1 Y. Zou: On Model Selection for Bayesian Networks and Sparse Logistic Regression. 58+61 pp. (Ph.D. Thesis)

A-2017-2 Y.-T. Hsieh: Exploring Hand-Based Haptic Interfaces for Mobile Interaction Design. 79+120 pp. (Ph.D. Thesis)

A-2017-3 D. Valenzuela: Algorithms and Data Structures for Sequence Analysis in the Pan-Genomic Era. 74+78 pp. (Ph.D. Thesis)

A-2017-4 A. Hellas: Retention in Introductory Programming. 68+88 pp. (Ph.D. Thesis)

A-2017-5 M. Du: Natural Language Processing System for Business Intelligence. 78+72 pp. (Ph.D. Thesis)

A-2017-6 A. Kuosmanen: Third-Generation RNA-Sequencing Analysis: Graph Alignment and Transcript Assembly with Long Reads. 64+69 pp. (Ph.D. Thesis)

A-2018-1 M. Nelimarkka: Performative Hybrid Interaction: Understanding Planned Events across Collocated and Mediated Interaction Spheres. 64+82 pp. (Ph.D. Thesis)

A-2018-2 E. Peltonen: Crowdsensed Mobile Data Analytics. 100+91 pp. (Ph.D. Thesis)

A-2018-3 O. Barral: Implicit Interaction with Textual Information using Physiological Signals. 72+145 pp. (Ph.D. Thesis)

A-2018-4 I. Kosunen: Exploring the Dynamics of the Biocybernetic Loop in Physiological Computing. 91+161 pp. (Ph.D. Thesis)

A-2018-5 J. Berg: Solving Optimization Problems via Maximum Satisfiability: Encodings and Re-Encodings. 86+102 pp. (Ph.D. Thesis)

A-2018-6 J. Pyykkö: Online Personalization in Exploratory Search. 101+63 pp. (Ph.D. Thesis)

A-2018-7 L. Pivovarova: Classification and Clustering in Media Monitoring: from Knowledge Engineering to Deep Learning. 78+56 pp. (Ph.D. Thesis)

A-2019-1 K. Salo: Modular Audio Platform for Youth Engagement in a Museum Context. 97+78 pp. (Ph.D. Thesis)

A-2019-2 A. Koski: On the Provisioning of Mission Critical Information Systems based on Public Tenders. 96+79 pp. (Ph.D. Thesis)

A-2019-3 A. Kantosalo: Human-Computer Co-Creativity - Designing, Evaluating and Modelling Computational Collaborators for Poetry Writing. 74+86 pp. (Ph.D. Thesis)

A-2019-4 O. Karkulahti: Understanding Social Media through Large Volume Measurements. 116 pp. (Ph.D. Thesis)

A-2019-5 S. Yaman: Initiating the Transition towards Continuous Experimentation: Empirical Studies with Software Development Teams and Practitioners. 81+90 pp. (Ph.D. Thesis)

A-2019-6 N. Mohan: Edge Computing Platforms and Protocols. 87+69 pp. (Ph.D. Thesis)

A-2019-7 I. Järvinen: Congestion Control and Active Queue Management During Flow Startup. 87+48 pp. (Ph.D. Thesis)

A-2019-8 J. Leinonen: Keystroke Data in Programming Courses. 56+53 pp. (Ph.D. Thesis)

A-2019-9 T. Talvitie: Counting and Sampling Directed Acyclic Graphs for Learning Bayesian Networks. 70+54 pp. (Ph.D. Thesis)

A-2019-10 J. Toivonen: Modeling and Learning Monomeric and Dimeric Transcription Factor Binding Motifs. 61+109 pp. (Ph.D. Thesis)

A-2019-11 S. Hemminki: Advances in Motion Sensing on Mobile Devices. 113+89 pp. (Ph.D. Thesis)

A-2019-12 P. Saikko: Implicit Hitting Set Algorithms for Constraint Optimization. 70+54 pp. (Ph.D. Thesis)